Quartz: Randomized Dual Coordinate Ascent with Arbitrary Sampling

Randomized Dual Coordinate Ascent with Arbitrary Sampling

Zheng Qu* Peter Richtárik† Tong Zhang‡

November 24, 2014

Abstract

We study the problem of minimizing the average of a large number of smooth convex functions penalized with a strongly convex regularizer. We propose and analyze a novel primal-dual method (Quartz) which at every iteration samples and updates a random subset of the dual variables, chosen according to an *arbitrary distribution*. In contrast to typical analysis, we directly bound the decrease of the primal-dual error (in expectation), without the need to first analyze the dual error. Depending on the choice of the sampling, we obtain efficient serial, parallel and distributed variants of the method. In the serial case, our bounds match the best known bounds for SDCA (both with uniform and importance sampling). With standard mini-batching, our bounds predict initial data-independent speedup as well as *additional data-driven speedup* which depends on spectral and sparsity properties of the data. We calculate theoretical speedup factors and find that they are excellent predictors of actual speedup in practice. Moreover, we illustrate that it is possible to design an efficient *mini-batch importance* sampling. The distributed variant of Quartz is the first distributed SDCA-like method with an analysis for non-separable data.

1 Introduction

In this paper we consider a primal-dual pair of structured convex optimization problems which has in several variants of varying degrees of generality attracted a lot of attention in the past few years in the machine learning and optimization communities [8, 9, 29, 27, 30, 28, 37].

1.1 The problem

Let $A_1, \dots, A_n$ be a collection of d-by-m real matrices and $\phi_1, \dots, \phi_n$ be $1/\gamma$-smooth convex functions from $\mathbb{R}^m$ to $\mathbb{R}$, where $\gamma > 0$. Further, let $g : \mathbb{R}^d \to \mathbb{R}$ be a 1-strongly convex function and $\lambda > 0$ a regularization parameter. We are interested in solving the following *primal* problem:

$$\min_{w=(w_1,\dots,w_d)\in\mathbb{R}^d} \left[P(w) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} \phi_i(A_i^\top w) + \lambda g(w) \right]. \tag{1}$$

In the machine learning context, matrices $\{A_i\}$ are interpreted as examples/samples, w is a (linear) predictor, function ϕ_i is the loss incurred by the predictor on example A_i, g is a regularizer,

*School of Mathematics, The University of Edinburgh, United Kingdom.

†School of Mathematics, The University of Edinburgh, United Kingdom.

‡Department of Statistics, Rutgers University, New Jersey, USA and Big Data Lab, Baidu Inc, China.

Acknowledgments: The first two authors would like to acknowledge support from the EPSRC Grant EP/K02325X/1, *Accelerated Coordinate Descent Methods for Big Data Optimization.*

λ is a regularization parameter and (1) is the *regularized empirical risk minimization* problem. However, above problem has many other applications outside machine learning. In this paper we are especially interested in problems where n is very big (millions, billions), and much larger than d. This is often the case in *big data* applications.

Let $g^* : \mathbb{R}^d \to \mathbb{R}$ be the convex conjugate[1] of g and for each i, let $\phi_i^* : \mathbb{R}^m \to \mathbb{R}$ be the convex conjugate of ϕ_i. Associated with the *primal problem* (1) is the Fenchel *dual problem*:

$$\max_{\alpha=(\alpha_1,\ldots,\alpha_n)\in\mathbb{R}^N=\mathbb{R}^{nm}} \quad \Big[D(\alpha) \stackrel{\text{def}}{=} -f(\alpha) - \psi(\alpha)\Big], \tag{2}$$

where $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{R}^N = \mathbb{R}^{nm}$ is obtained by stacking dual variables (blocks) $\alpha_i \in \mathbb{R}^m$, $i = 1, \ldots, n$, on top of each other and functions f and ψ are defined by

$$f(\alpha) \stackrel{\text{def}}{=} \lambda g^*\left(\frac{1}{\lambda n}\sum_{i=1}^n A_i\alpha_i\right), \tag{3}$$

$$\psi(\alpha) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^n \phi_i^*(-\alpha_i). \tag{4}$$

Note that f is convex and smooth and ψ is strongly convex and block separable.

1.2 Contributions

We now briefly list the main contributions of this work.

Quartz. We propose a new algorithm, which we call Quartz[2], for simultaneously solving the primal (1) and dual (2) problems. On the dual side, at each iteration our method selects and updates a *random subset (sampling)* $\hat{S} \subseteq \{1, \ldots, n\}$ of the dual variables/blocks. We assume that these sets are i.i.d. throughout the iterations. However, *we do not impose any additional assumptions on the distribution* apart from the necessary requirement that each block $i \in [n]$ needs to be chosen with a positive probability: $p_i \stackrel{\text{def}}{=} \mathbb{P}(i \in \hat{S}) > 0$. Quartz is the first SDCA-like method analyzed for an *arbitrary sampling*. The dual updates are then used to perform an update to the primal variable w and the process is repeated. Our primal updates are different (less aggressive) from those used in SDCA [29] and Prox-SDCA [27].

Main result. We prove that starting from an initial pair (w^0, α^0), Quartz finds a pair (w, α) for which $P(w) - D(\alpha) \leq \epsilon$ (in expectation) in at most

$$\max_i \left(\frac{1}{p_i} + \frac{v_i}{p_i\lambda\gamma n}\right)\log\left(\frac{P(w^0) - D(\alpha^0)}{\epsilon}\right) \tag{5}$$

[1] In this paper, the convex (Fenchel) conjugate of a function $\xi : \mathbb{R}^k \to \mathbb{R}$ is the function $\xi^* : \mathbb{R}^k \to \mathbb{R}$ defined by $\xi^*(u) = \sup_{\|s\|=1}\{s^\top u - \xi(s)\}$, where $\|\cdot\|$ is the L2 norm.

[2] Strange as it may seem, this algorithm name appeared to one of the authors of this paper in a dream. According to Wikipedia: "Quartz is the second most abundant mineral in the Earth's continental crust. There are many different varieties of quartz, several of which are semi-precious gemstones." Our method also comes in many variants. It later came as a surprise to the authors that the name could be interpreted as QU And Richtárik and Tong Zhang. Whether the subconscious mind of the sleeping coauthor who dreamed up the name knew about this connection or not is not known.

iterations. The parameters $v_1, \dots, v_n$ are assumed to satisfy the following ESO (expected separable overapproximation) inequality:

$$\mathbb{E}_{\hat{S}}\left[\left\|\sum_{i\in\hat{S}} A_i h_i\right\|^2\right] \leq \sum_{i=1}^n p_i v_i \|h_i\|^2. \tag{6}$$

Moreover, the parameters are needed to run the method (they determine stepsizes), and hence it is critical that they can be cheaply computed before the method starts. As we will show, for many samplings of interest this can be done in time required to read the data $\{A_i\}$. We wish to point out that (6) always holds for *some* parameters $\{v_i\}$. Indeed, the left hand side is a quadratic function of h and hence the inequality holds for large-enough v_i. Having said that, the size of these parameters directly influences the complexity, and hence one would want to obtain as tight bounds as possible.

Arbitrary sampling. As described above, Quartz uses an *arbitrary sampling* for picking the dual variables to be updated in each iteration. To the best of our knowledge, only a single paper exists in the literature where a stochastic method using an arbitrary sampling was analyzed: the NSync method of Richtárik and Takáč [22] (for unconstrained minimization of a strongly convex function). Assumption (6) was for the first time introduced there (in a more general form; we are using it here in the special case of a quadratic function). However, NSync is not a primal-dual method. Besides NSync, the closest works to ours in terms of the generality of the sampling are the PCDM algorithm of Richtárik and Takáč [23], SPCDM method of Fercoq and Richtárik [7] and the APPROX method of Fercoq and Richtárik [6]. All these are randomized coordinate descent methods, and all were analyzed for arbitrary *uniform* samplings (i.e., samplings satisfying $\mathbb{P}(i \in \hat{S}) = \mathbb{P}(i' \in \hat{S})$ for all $i, i' \in [n]$). Again, none of these methods were analyzed in a primal-dual framework.

Direct primal-dual analysis. Virtually all methods for solving (1) by performing stochastic steps in the dual (2), such as SDCA [29], SDCA for SVM dual [30], ProxSDCA [27], ASDCA [28] and APCG [15], are analyzed by first establishing dual convergence and then proving that the duality gap is bounded by the dual residual. The SPDC method of Zhang and Xiao [36], which is a stochastic coordinate update variant of the Chambolle-Pock method [3], is an exception. Our analysis is novel, and *directly primal-dual* in nature. As a result, our proof is more direct, and the logarithmic term in our bound has a simpler form.

Flexibility: many important variants. Our method is very flexible: by specializing it to specific samplings, we obtain numerous variants, some similar (but not identical) to existing methods in the literature, and some very new and of significance to big data optimization.

- **Serial uniform sampling.** If $\hat{S}$ always picks a single block, uniformly at random ($p_i = 1/n$), then the dual updates of Quartz are similar to those of SDCA [29] and Prox-SDCA [27]. The leading term in the complexity bound (5) becomes $n + \max_i \lambda_{\max}(A_i^\top A_i)/(\lambda\gamma)$, which matches the bounds obtained in these papers. However, our logarithmic term is simpler.
- **Serial optimal sampling (importance sampling).** If $\hat{S}$ always picks a single block, with p_i chosen so as to minimize the complexity bound (5), we obtain the same *importance sampling* as that recently used in the IProx-SDCA method [37]. Our bound becomes $n + (\frac{1}{n}\sum_i \lambda_{\max}(A_i^\top A_i))/(\lambda\gamma)$, which matches the bound in [37]. Again, our logarithmic term is better.

- **τ-nice sampling.** If we now let $\hat{S}$ be a random subset of $[n]$ of size τ chosen uniformly at random (this sampling is called τ-nice in [23]), we obtain a mini-batch (parallel) variant of Quartz. There are only a handful of primal-dual stochastic methods which use mini-batching. The first such method was a mini-batch version of SDCA specialized to training $L2$-regularized linear SVMs with hinge loss [30]. Besides this, two accelerated mini-batch methods have been recently proposed: ASDCA of Shalev-Shwartz and Zhang [28] and SPDC of Zhang and Xiao [36]. The complexity bound of Quartz specialized to the τ-nice sampling is different, and despite Quartz not being an accelerated method, and can be better in certain regimes (we will do a detailed comparison in Section 4).

- **Distributed sampling.** To the best of our knowledge, no other samplings than those described above were used in stochastic primal-dual methods. However, there are many additional interesting samplings proposed for randomized coordinate descent, but never applied to the primal-dual framework. For instance, we can use the *distributed sampling* which led to the development of the Hydra algorithm [21] (distributed coordinate descent) and its accelerated variant Hydra^2 (Hydra squared) [5]. Using this sampling, Quartz can be efficiently implemented in a distributed environment (partition the examples across the nodes of a cluster, and let each node in each iteration update a random subset of variables corresponding to the examples it owns).

- **Product sampling.** We describe a novel sampling, which we call *product sampling*, that can be *both non-serial and non-uniform*. This is the first time such a sampling has been described and and a SDCA-like method using it analyzed. For suitable data (if the examples can be partitioned into several groups no two of which share a feature), this sampling can lead to linear or nearly linear speedup when compared to the serial uniform sampling.

- **Other samplings.** While we develop the analysis of Quartz for an arbitrary sampling, we do not compute the ESO parameters $\{v_i\}$ for any other samplings in this paper. However, there are several other interesting choices. We refer the reader to [23] and [22] for further examples of uniform and non-uniform samplings, respectively. All that must be done for any new $\hat{S}$ is to find parameters v_i for which (6) holds, and the complexity of the new variant of Quartz is given by (5).

Further data-driven speedup. Existing mini-batch stochastic primal-dual methods achieve linear speedup up to a certain mini-batch size which depends on n, λ and γ. Quartz obtains this data-independent speedup, but also obtains *further data-driven speedup*. This is caused by the fact that Quartz uses more aggressive dual stepsizes, informed by the data through the ESO parameters $\{v_i\}$. The smaller these constants, the better speedup. For instance, we will show that higher data sparsity leads to smaller $\{v_i\}$ and hence to better speedup. To illustrate this, consider the τ-nice sampling (hence, $p_i = \tau/n$ for all i) and the extreme case of perfectly sparse data (each feature $j \in [d]$ appearing in a single example A_i). Then (6) holds with $v_i = \lambda_{\max}(A_i^\top A_i)$ for all i, and hence the leading term in (5) becomes $n/\tau + \max_i \lambda_{\max}(A_i^\top A_i)/(\gamma\lambda\tau)$, predicting *perfect speedup* in the mini-batch size τ. We derive *theoretical speedup factors* and show that these are excellent predictors of actual behavior of the method in an implementation. This was previously observed for the PCDM method [23] (which is not primal-dual).

Quartz vs purely primal and purely dual methods. In the special case when $\hat{S}$ is the serial uniform sampling, the complexity of Quartz is similar to the bounds recently obtained by several purely primal stochastic and semi-stochastic gradient methods (all having reduced variance of the gradient estimate) such as SAG [25], SVRG [11], S2GD [14], SAGA [4], mS2GD [12] and MISO [16]. In the case of serial optimal sampling, relevant purely primal methods with similar guarantees are ProxSVRG [33] and S2CD [13]. A mini-batch primal method, mS2GD, was analyzed in [12], achieving a similar bound to Quartz specialized to the τ-nice sampling. Purely dual (stochastic coordinate descent) methods with similar bounds to Quartz for both the serial uniform and serial optimal sampling, for problems of varying similarity and generality when compared to (2), include SCD [26], RCDM [19], UCDC/RCDC [24], ICD [32] and RCD [18]. These methods were then generalized to the τ-nice sampling in SHOTGUN [2], further generalized to arbitrary uniform samplings in PCDM [23], SPCDM [7], APPROX [6] (which is an accelerated method) and to arbitrary (even nonuniform) samplings in NSync [22]. Another accelerated method, BOOM, was proposed in [17]. Distributed randomized coordinate descent methods with purely dual analysis include Hydra [21] and Hydra^2 [5] (accelerated variant of Hydra). Quartz specialized to the distributed sampling achieves the same rate as Hydra, but for both the primal and dual problems simultaneously.

General problem. We consider the problem (1) (and consequently, the associated dual) in a rather general form; most existing primal-dual methods focus on the case when g is a quadratic (e.g., [29, 28]) or $m = 1$ (e.g., [36]). Lower bounds for a variant of problem (1) were recently established by Agarwal and Bottou [1].

1.3 Outline

In Section 2 we describe the algorithm and show that it admits a natural interpretation in terms of Fenchel duality. We also outline the similarities and differences of the primal and dual update steps with SDCA-like methods. In Section 3 we show how parameters $\{v_i\}$ satisfying the ESO inequality (6) can be computed for several selected samplings. We then proceed to Section 4 where we state the main result, specialize it to some of the samplings discussed in Section 3. Sections 5 and 6 deal with Quartz specialized to the τ-nice and distributed sampling, respectively. We also give detailed comparison of our results with existing results for related primal-dual stochastic methods existing in the literature, and analyze theoretical speedup factors. We then provide the proof of the main complexity result in Section 7. In Section 8 we perform numerical experiments on the problem of training L_2-regularized linear support vector machine with square and smoothed hinge loss with real datasets. Finally, in Section 9 we conclude.

2 The Quartz Algorithm

In this section we describe our method (Algorithm 1).

2.1 Preliminaries

The most important parameter of Quartz is a random sampling $\hat{S}$ of the dual variables $[n] = \{1, 2, \dots, n\}$. That is, $\hat{S}$ is a random subset of $[n]$, or more precisely, a random set-valued mapping with values being the subsets of $[n]$. In order to guarantee that each block (dual variable) has a chance to get updated by the method, we necessarily need to make the following assumption.

Assumption 1 (Proper sampling) *$\hat{S}$ is a proper sampling. That is,*

$$p_i \stackrel{def}{=} \mathbb{P}(i \in \hat{S}) > 0, \qquad i \in [n]. \tag{7}$$

However, we shall not make any other assumption on $\hat{S}$. Prior to running the algorithm, we compute positive constants $v_1, \dots, v_n$ satisfying (6)—such constants always exist—as these are used to define the stepsize parameter θ used throughout:

$$\theta = \min_i \frac{p_i \lambda \gamma n}{v_i + \lambda \gamma n}. \tag{8}$$

We shall show how this parameter can be computed for various samplings in Section 3. Let us now formalize the notions of $(1/\gamma)$-smoothness and strong convexity.

Assumption 2 (Loss) *For each $i \in [n]$, the loss function $\phi_i : \mathbb{R}^m \to \mathbb{R}$ is convex, differentiable and has $(1/\gamma)$-Lipschitz continuous gradient with respect to the L2 norm, where γ is a positive constant:*

$$\|\nabla \phi_i(x) - \nabla \phi_i(y)\| \leq \frac{1}{\gamma}\|x - y\|, \quad x, y \in \mathbb{R}^m.$$

For brevity, the last property is often called $(1/\gamma)$-smoothness.

It follows that ϕ_i^* is γ-strongly convex.

Assumption 3 (Regularizer) *The regularizer $g : \mathbb{R}^d \to \mathbb{R}$ is 1-strongly convex. That is,*

$$g(w) \geq g(w') + \langle \nabla g(w'), w - w' \rangle + \tfrac{1}{2}\|w - w'\|^2, \quad w, w' \in \mathbb{R}^d,$$

where $\nabla g(w')$ is a subgradient of g at w'.

It follows that g^* is 1-smooth.

2.2 Description of the method

Quartz starts with an initial pair of primal and dual vectors (w^0, α^0). Given w^{t-1} and α^{t-1}, the method maintains the vector

$$\bar{\alpha}^{t-1} = \frac{1}{\lambda n} \sum_{i=1}^{n} A_i \alpha_i^{t-1}. \tag{9}$$

Initially this is computed from scratch, and subsequently it is maintained in an efficient manner at the end of each iteration.

Let us now describe how the vectors w^t and α^t are computed. Quartz first updates the primal vector w^t by setting it to a *convex combination* of the previous value w^{t-1} and $\nabla g^*(\bar{\alpha}^{t-1})$:

$$w^t = (1 - \theta) w^{t-1} + \theta \nabla g^*(\bar{\alpha}^{t-1}). \tag{10}$$

We then proceed to select, and subsequently update, a random subset $S_t \subseteq [n]$ of the dual variables, independently from the sets drawn in previous iterations, and following the distribution of $\hat{S}$. Clearly, there are many ways in which the distribution of $\hat{S}$ can be chosen, leading the numerous variants of Quartz. We shall describe some of them in Section 3. We allow two options for the actual computation of the dual updates. Once the dual variables are updated, the vector $\bar{\alpha}^t$ is updated in an efficient manner so that (9) holds. The entire process is repeated.

Algorithm 1 Quartz

Parameters: proper random sampling $\hat{S}$ and a positive vector $v \in \mathbb{R}^n$
Initialization: Choose $\alpha^0 \in \mathbb{R}^N$ and $w^0 \in \mathbb{R}^d$
Set $p_i = \mathbb{P}(i \in \hat{S})$, $\theta = \min_i \frac{p_i \lambda\gamma n}{v_i + \lambda\gamma n}$ and $\bar{\alpha}^0 = \frac{1}{\lambda n}\sum_{i=1}^n A_i \alpha_i^0$
for $t \geq 1$ **do**
  $w^t = (1-\theta)w^{t-1} + \theta \nabla g^*(\bar{\alpha}^{t-1})$
  $\alpha^t = \alpha^{t-1}$
  Generate a random set $S_t \subseteq [n]$, following the distribution of $\hat{S}$
  for $i \in S_t$ **do**
    Calculate $\Delta\alpha_i^t$ using one of the following options:
      Option I :
      $\Delta\alpha_i^t = \arg\max_{\Delta \in \mathbb{R}^m} \left[-\phi_i^*(-(\alpha_i^{t-1} + \Delta)) - \nabla g^*(\bar{\alpha}^{t-1})^\top A_i \Delta - \frac{v_i \|\Delta\|^2}{2\lambda n} \right]$
      Option II :
      $\Delta\alpha_i^t = -\theta p_i^{-1} \alpha_i^{t-1} - \theta p_i^{-1} \nabla \phi_i(A_i^\top w^t)$
    $\alpha_i^t = \alpha_i^{t-1} + \Delta\alpha_i^t$
  end for
  $\bar{\alpha}^t = \bar{\alpha}^{t-1} + (\lambda n)^{-1} \sum_{i \in S_t} A_i \Delta\alpha_i^t$
end for
Output: w^t, α^t

Fenchel duality interpretation. Quartz has a natural interpretation in terms of Fenchel duality. Fix a primal-dual pair of vectors $(w, \alpha) \in \mathbb{R}^d \times \mathbb{R}^N$ and define $\bar{\alpha} = \frac{1}{\lambda n}\sum_{i=1}^n A_i \alpha_i$. The duality gap for the pair (w, α) can be decomposed as follows:

$$\begin{aligned} P(w) - D(\alpha) &\overset{(1)+(2)}{=} \lambda\left(g(w) + g^*\left(\bar{\alpha}\right)\right) + \frac{1}{n}\sum_{i=1}^n \phi_i(A_i^\top w) + \phi_i^*(-\alpha_i) \\ &= \lambda(\underbrace{g(w) + g^*\left(\bar{\alpha}\right) - \langle w, \bar{\alpha}\rangle}_{GAP_g(w,\alpha)}) + \frac{1}{n}\sum_{i=1}^n \underbrace{\phi_i(A_i^\top w) + \phi_i^*(-\alpha_i) + \langle A_i^\top w, \alpha_i\rangle}_{GAP_{\phi_i}(w,\alpha_i)}. \end{aligned}$$

By Fenchel-Young inequality, $GAP_g(w,\alpha) \geq 0$ and $GAP_{\phi_i}(w,\alpha_i) \geq 0$ for all i, which proves weak duality for the problems (1) and (2), i.e., $P(w) \geq D(\alpha)$. The pair (w,α) is optimal when both GAP_g and GAP_{ϕ_i} for all i are zero. It is known that this happens precisely when the following optimality conditions hold:

$$w = \nabla g^*(\bar{\alpha}), \tag{11}$$

$$\alpha_i = -\nabla \phi_i(A_i^\top w), \quad \forall i \in [n]. \tag{12}$$

We will now interpret the primal and dual steps of Quartz in terms of the above discussion. At iteration t we first set the primal variable w^t to a convex combination of its current value w^{t-1} and a value that would set GAP_g to zero: see (10). Hence, our primal update is not as aggressive as that of Prox-SDCA. This is followed by adjusting the dual variables corresponding to a randomly chosen set of examples S_t. Under Option II, for each example $i \in S_t$, the i-th dual variable α_i^t is set to a convex combination of its current value α_i^{t-1} and the value that would set GAP_{ϕ_i} to zero:

$$\alpha_i^t = \left(1 - \frac{\theta}{p_i}\right)\alpha_i^{t-1} + \frac{\theta}{p_i}\left(-\nabla\phi_i(A_i^\top w^t)\right).$$

Quartz vs Prox-SDCA. In the special case when $\hat{S}$ is the serial uniform sampling (i.e., $p_i = 1/n$ for all $i \in [n]$), Quartz can be compared to Proximal Stochastic Dual Coordinate Ascent (Prox-SDCA) [28, 29]. Indeed, if Option I is always used in Quartz, then the dual update of α^t in Quartz is exactly the same as the dual update of Prox-SDCA (using Option I). In this case, the difference between our method and Prox-SDCA lies in the update of the primal variable w^t: while Quartz performs the update (10), Prox-SDCA (see also [34, 10]) performs the more aggressive update $w^t = \nabla g^*(\bar{\alpha}^{t-1})$.

3 Expected Separable Overapproximation

For the sake of brevity, it will be convenient to establish some notation. Let $A = [A_1, \dots, A_n] \in \mathbb{R}^{d\times N} = \mathbb{R}^{d\times nm}$ be the block matrix with blocks $A_i \in \mathbb{R}^{d\times m}$. Further, let A_{ji} be the j-th row of A_i. Likewise, for $h \in \mathbb{R}^N$ we will write $h = (h_1, \dots, h_n)$, where $h_i \in \mathbb{R}^m$, so that $Ah = \sum_{i=1}^n A_i h_i$. For a vector of positive weights $w \in \mathbb{R}^n$, we define a weighted Euclidean norm in $\mathbb{R}^N$ by

$$\|h\|_w^2 \stackrel{\text{def}}{=} \sum_{i=1}^n w_i \|h_i\|^2, \tag{13}$$

where $\|\cdot\|$ is the standard Euclidean norm on $\mathbb{R}^m$. For $S \subset [n] \stackrel{\text{def}}{=} \{1, \dots, n\}$ and $h \in \mathbb{R}^N$ we use the notation $h_{[S]}$ to denote the vector in $\mathbb{R}^N$ coinciding with h for blocks $i \in S$ and zero elsewhere:

$$(h_{[S]})_i = \begin{cases} h_i, & \text{if } i \in S, \\ 0, & \text{otherwise.} \end{cases}$$

With this notation, we have

$$Ah_{[S]} = \sum_{i\in S} A_i h_i. \tag{14}$$

As mentioned before, in our analysis we require that the random sampling $\hat{S}$ and the positive vector $v \in \mathbb{R}^n$ used in Quartz satisfy inequality (6). We shall now formalize this as an assumption, using the compact notation established above.

Assumption 4 (ESO) *The following inequality holds for all $h \in \mathbb{R}^N$:*

$$\mathbb{E}[\|Ah_{[\hat{S}]}\|^2] \le \|h\|_{p\cdot v}^2, \tag{15}$$

where $p = (p_1, \dots, p_n)$ is defined in (7), $v = (v_1, \dots, v_n) > 0$ and $p \cdot v = (p_1 v_1, \dots, p_n v_n) \in \mathbb{R}^n$.

Note that for any proper sampling $\hat{S}$, there must *exist* vector $v > 0$ satisfying Assumption 4. Hence, this is an assumption that such a vector v is *readily available*. Indeed, the term on the left is a finite average of convex quadratic functions of h, and hence is a convex quadratic. Moreover, we can write

$$\mathbb{E}[\|Ah_{[\hat{S}]}\|^2] = \mathbb{E}[h_{[\hat{S}]}^\top A^\top A h_{[\hat{S}]}] = h^\top \left(P \circ A^\top A\right) h,$$

where $\circ$ denotes the Hadamard (component-wise) product of matrices and $P \in \mathbb{R}^{N\times N}$ is an n-by-n block matrix with block (i, j) equal to $\mathbb{P}(i \in \hat{S}, j \in \hat{S})1_m$, with 1_m being the m-by-m matrix of all ones. Hence (15) merely means to upper bound the matrix $P \circ A^\top A$ by an n-by-n block diagonal

matrix $D = D_{p,v}$, the i-th block of which is equal to $p_i v_i I_m$ with I_m being the m-by-m identity matrix. There is an infinite number of ways how this can be done (in theory). Indeed, for any proper sampling $\hat{S}$ and *any* positive $w \in \mathbb{R}^n$, (15) holds with $v = tw$, where

$$t = \lambda_{\max}\left(D_{p,w}^{-1/2}(P \circ A^T A)D_{p,w}^{-1/2}\right),$$

since then $P \circ A^\top A \preceq tD_{p,w} = D_{p,v}$.

In practice, and especially in the big data setting when n is very large, computing v by solving an eigenvalue problem with an $N \times N$ matrix (recall that $N = nm$) will be either inefficient or impossible. It is therefore important that a "good" (i.e., small), albeit perhaps suboptimal v can be identified *cheaply.* In all the cases we consider in this paper, the identification of v can be done during the time the data is being read; or in time roughly equal to a single pass through the data matrix A.

In the special case of uniform[3] samplings but for arbitrary smooth functions (and not just quadratics; which is all we need here), inequality (15) was introduced and studied by Richtárik and Takáč [23], in the context of complexity analysis of (non primal-dual) parallel block coordinate descent methods. A variant of ESO for arbitrary (possibly nonuniform) samplings was introduced in [22]; and to the best of our knowledge that is the only work analyzing a stochastic coordinate descent method which uses an arbitrary sampling. However, NSync is not a primal-dual method and applies to a different problem (unconstrained minimization of a smooth strongly convex function). Besides [23, 22], ESO inequalities were further studied in [30, 31, 7, 21, 6, 5, 12].

3.1 Serial samplings

The most studied sampling in literature on stochastic optimization is the *serial sampling*, which corresponds to the selection of a single block $i \in [n]$. That is, $|\hat{S}| = 1$ with probability 1. The name "serial" is pointing to the fact that a method using such a sampling will typically be a serial (as opposed to being parallel) method; updating a single block (dual variable) at a time.

A serial sampling is uniquely characterized by the vector of probabilities $p = (p_1, \dots, p_n)$, where p_i is defined by (7). It turns out that we can find a vector $v > 0$ for which (15) holds for *any* serial sampling, *independently of its distribution* given by p.

Lemma 5 *If $\hat{S}$ is a serial sampling (i.e., if $|\hat{S}| = 1$ with probability 1), then Assumption 4 is satisfied for*

$$v_i = \lambda_{\max}(A_i^\top A_i), \quad i \in [n]. \tag{16}$$

Proof Note that for any $h \in \mathbb{R}^N$,

$$\mathbb{E}[\|Ah_{[\hat{S}]}\|^2] = \sum_{i=1}^n p_i \|Ah_{[\{i\}]}\|^2 \overset{(14)}{=} \sum_{i=1}^n p_i (h_i A_i^\top A_i h_i) \leq \sum_{i=1}^n p_i \lambda_{\max}(A_i^\top A_i)\|h_i\|^2 \overset{(13)}{=} \|h\|_{p\cdot v}^2.$$

■

[3] A sampling $\hat{S}$ is uniform if $p_i = p_j$ for all i, j. It is easy to see that then, necessarily, $p_i = \mathbb{E}[|\hat{S}|]/n$ for all i. The ESO inequality studied in [23] is of the form: $\mathbb{E}[\xi(x + h_{[\hat{S}]})] \leq \xi(x) + \frac{\mathbb{E}[|\hat{S}|]}{n}\left(\langle \nabla \xi(x), h\rangle + \frac{1}{2}\|h\|_v^2\right)$. In the case of uniform sampling, $x = 0$ and $\xi(h) = \frac{1}{2}\|Ah\|^2$, we recover (15).

Note that v_i is the largest eigenvalue of an m-by-m matrix. If m is relatively small (and in many machine learning applications one has $m = 1$; as examples are usually vectors and not matrices), then the cost of computing v_i is small. If $m = 1$, then v_i is simply the squared Euclidean norm of the vector A_i, and hence one can compute all of these parameters in one pass through the data (e.g., during loading to memory).

3.2 Parallel (τ-nice) sampling

We now consider $\hat{S}$ which selects subsets of $[n]$ of cardinality τ, uniformly at random. In the terminology established in [23], such $\hat{S}$ is called τ-nice. This sampling satisfies $p_i = p_j$ for all $i, j \in [n]$; and hence it is uniform.

This sampling is well suited for parallel computing. Indeed, Quartz could be implemented as follows. If we have τ processors available, then at the beginning of iteration t we can assign each block (dual variable) in S_t to a dedicated processor. The processor assigned to i would then compute $\Delta\alpha_i^t$ and apply the update. If all processors have fast access to the memory where all the data is stored, as is the case in a shared-memory multicore workstation, then this way of assigning workload to the individual processors does not cause any major problems. Depending on the particular computer architecture and the size m of the blocks (which will influence processing time), it may be more efficient to chose τ to be a multiple of the number of processors available, in which case in each iteration every processor updates more than one block.

The following lemma gives a closed-form formula for parameters $\{v_i\}$ for which the ESO inequality holds.

Lemma 6 (compare with [6]) *If $\hat{S}$ is a τ-nice sampling, then Assumption 4 is satisfied for*

$$v_i = \lambda_{\max}\left(\sum_{j=1}^{d}\left(1 + \frac{(\omega_j - 1)(\tau - 1)}{n - 1}\right) A_{ji}^\top A_{ji}\right), \quad i \in [n], \tag{17}$$

where for each $j \in [d]$, ω_j is the number of nonzero blocks in the j-th row of matrix A, i.e.,

$$\omega_j \overset{def}{=} |\{i \in [n] : A_{ji} \neq 0\}|, \qquad j \in [d]. \tag{18}$$

Proof In the $m = 1$ case the result follows from Theorem 1 in [6]. Extension to the $m > 1$ case is straightforward. ■

Note that v_i is the largest eigenvalue of an m-by-m matrix which is formed as the sum of d rank-one matrices. The formation of all of these n matrices takes time proportional to the number of nonzeros in A (if the data is stored in a sparse format). Constants $\{\omega_j\}$ can be computed by scanning the data once (e.g., during loading-to-memory phase). Finally, one must compute n eigenvalue problems for matrices of size $m \times m$. In most applications, $m = 1$, so there is no more work to be done. If $m > 1$, the cost of computing these eigenvalues would be small.

While for $\tau = 1$ it was easy to find parameters $\{v_i\}$ for any sampling (and hence, as we will see, it will be easy to find an optimal sampling), this is not the case in the $\tau > 1$ case. The task is in general a difficult optimization problem. For some work in this direction we refer the reader to [22].

3.3 Product sampling

In this section we give an example of a sampling $\hat{S}$ which can be both non-uniform and non-serial (i.e., for which $\mathbb{P}(|\hat{S}| = 1) \neq 1$). We make the following *group separability assumption*: there is a partition $X_1, \ldots, X_\tau$ of $[n]$ according to which the examples $\{A_i\}$ can be partitioned into τ groups such that no feature is shared by any two examples belonging to different groups.

Consider the following example with $m = 1, n = 5$ and $d = 4$:

$$A = [A_1, A_2, A_3, A_4, A_5] = \begin{pmatrix} 0 & 0 & 6 & 4 & 9 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 1 \\ 1 & 8 & 0 & 0 & 0 \end{pmatrix}$$

If we choose $\tau = 2$ and $X_1 = \{1, 2\}, X_2 = \{3, 4, 5\}$, then no row of A has a nonzero in both a column belonging to X_1 and a column belonging to X_2.

With each $i \in [n]$ we now associate $l_i \in [\tau]$ such that $i \in X_{l_i}$ and define:

$$\mathcal{S} \stackrel{\text{def}}{=} X_1 \times \cdots \times X_\tau.$$

The *product sampling* $\hat{S}$ is obtained by choosing $S \in \mathcal{S}$, uniformly at random; that is, via:

$$\mathbb{P}(\hat{S} = S) = \frac{1}{|\mathcal{S}|} = \frac{1}{\prod_{l=1}^{\tau} |X_l|}, \quad S \in \mathcal{S}. \tag{19}$$

Then $\hat{S}$ is proper and

$$p_i \stackrel{\text{def}}{=} \mathbb{P}(i \in \hat{S}) = \frac{\prod_{l \neq l_i} |X_l|}{|\mathcal{S}|} \stackrel{(19)}{=} \frac{1}{|X_{l_i}|}, \qquad i \in [n]. \tag{20}$$

Hence the sampling is nonuniform as long as not all of the sets X_l have the same cardinality. We next show that the product sampling $\hat{S}$ defined as above allows the same stepsize parameter v_i as the serial uniform sampling.

Lemma 7 *Under the group separability assumption, Assumption 4 is satisfied for the product sampling $\hat{S}$ and*

$$v_i = \lambda_{\max}(A_i^\top A_i), \quad i \in [n].$$

Proof For each $j \in [d]$, denote by $A_{j:}$ the j-th row of the matrix A and Ω_j the column index set of nonzero blocks in $A_{j:}$: $\Omega_j \stackrel{\text{def}}{=} \{i \in [n] : A_{ji} \neq 0\}$. For each $l \in [\tau]$, define:

$$J_l \stackrel{\text{def}}{=} \{j \in [d] : \Omega_j \subset X_l\}. \tag{21}$$

In words, J_l is the set of features associated with the examples in X_l. By the group separability assumption, $J_1, \ldots, J_\tau$ forms a partition of $[d]$, namely,

$$\bigcup_{l=1}^{\tau} J_l = [d]; \;\; J_k \cap J_l = \emptyset, \;\; \forall k \neq l \in [\tau]. \tag{22}$$

Thus,

$$A^\top A = \sum_{j=1}^{d} A_{j:}^\top A_{j:} \overset{(22)}{=} \sum_{l=1}^{\tau} \sum_{j \in J_l} A_{j:}^\top A_{j:}. \tag{23}$$

Now fix $l \in [\tau]$ and $j \in J_l$. For any $h \in \mathbb{R}^N$ we have:

$$\mathbb{E}[h_{[\hat{S}]} A_{j:}^\top A_{j:} h_{[\hat{S}]}] = \sum_{i,i' \in [n]} h_i^\top A_{ji}^\top A_{ji'} h_{i'} \mathbb{P}(i \in \hat{S}, i' \in \hat{S}) = \sum_{i,i' \in \Omega_j} h_i^\top A_{ji}^\top A_{ji'} h_{i'} \mathbb{P}(i \in \hat{S}, i' \in \hat{S}).$$

Since $X_1, \ldots, X_\tau$ forms a partition of $[n]$, then any two indexes belonging to the same subset X_l will never be selected simultaneously in $\hat{S}$, i.e.,

$$\mathbb{P}(i \in \hat{S}, i' \in \hat{S}) = \begin{cases} p_i & \text{if } \ i = i' \\ 0 & \text{if } \ i \neq i' \end{cases}, \quad \forall i, i' \in X_l.$$

Therefore,

$$\mathbb{E}[h_{[\hat{S}]} A_{j:}^\top A_{j:} h_{[\hat{S}]}] = \sum_{i \in \Omega_j} h_i^\top A_{ji}^\top A_{ji} h_i p_i = \sum_{i=1}^{n} h_i^\top A_{ji}^\top A_{ji} h_i p_i. \tag{24}$$

It follows from (23) and (24) that:

$$\mathbb{E}[\|A h_{[\hat{S}]}\|^2] = \mathbb{E}[h_{[\hat{S}]} A^\top A h_{[\hat{S}]}] = \sum_{l=1}^{\tau} \sum_{j \in J_l} \mathbb{E}[h_{[\hat{S}]} A_{j:}^\top A_{j:} h_{[\hat{S}]}] = \sum_{l=1}^{\tau} \sum_{j \in J_l} \sum_{i=1}^{n} h_i^\top A_{ji}^\top A_{ji} h_i p_i. \tag{25}$$

Hence, $\mathbb{E}[\|A h_{[\hat{S}]}\|^2] \overset{(22)}{=} \sum_{j=1}^{d} \sum_{i=1}^{n} h_i^\top A_{ji}^\top A_{ji} h_i p_i \leq \sum_{i=1}^{n} \lambda_{\max}(A_i^\top A_i) h_i^\top h_i p_i = \|h\|_{p \cdot v}^2$. ■

3.4 Distributed sampling

We now describe a sampling which is particularly suitable for a *distributed implementation of Quartz*. This sampling was first proposed in [21] and later used in [5], where the distributed coordinate descent algorithm Hydra and its accelerated variant Hydra^2 were proposed and analyzed, respectively. Both methods were shown to be able to scale up to huge problem sizes (tests were performed on problem sizes of several TB; and up 50 billion dual variables in size).

Consider a distributed computing environment with c nodes/computers. For simplicity, assume that n is an integer multiple of c and let the blocks $\{1, 2, \ldots, n\}$ be partitioned into c sets of equal size: $\mathcal{P}_1$, $\mathcal{P}_2$, $\ldots$, $\mathcal{P}_c$. We assign partition $\mathcal{P}_l$ to node l. The data $A_1, \ldots, A_n$ and the dual variables (blocks) $\alpha_1, \ldots, \alpha_n$ are partitioned accordingly and stored on the respective nodes.

At each iteration, all nodes $l \in \{1, \ldots, c\}$ in parallel pick a subset $\hat{S}_l$ of τ dual variables from those they own, i.e., from $\mathcal{P}_l$, uniformly at random. That is, each node locally performs a τ-nice sampling, independently from the other nodes. Node l computes the updates to the dual variables α_i corresponding to $i \in S_l$, and locally stores them. Hence, in a single distributed iteration, Quartz updates the dual variables belonging to the set $\hat{S} \overset{\text{def}}{=} \cup_{l=1}^{c} \hat{S}_l$. This defines a sampling, which we will call (c, τ)-*distributed sampling*.

Of course, there are other important considerations pertaining to the distributed implementation of Quartz, but we do not discuss them here as the focus of this section is on the sampling. However, it is possible to design a distributed communication protocol for the update of the primal variable.

The following result gives a formula for admissible parameters $\{v_i\}$.

Lemma 8 (compare with [5]) *If $\hat{S}$ is a (c,τ)-distributed sampling, then Assumption 4 is satisfied for*

$$v_i = \lambda_{\max}\left(\sum_{j=1}^{d}\left(1+\frac{(\tau-1)(\omega_j-1)}{\max\left\{\frac{n}{c}-1,1\right\}}+\left(\frac{\tau c}{n}-\frac{\tau-1}{\max\{\frac{n}{c}-1,1\}}\right)\frac{\omega_j'-1}{\omega_j'}\omega_j\right)A_{ji}^\top A_{ji}\right), \quad i\in[n], \tag{26}$$

where ω_j is the number of nonzero blocks in the j-th row of the matrix A, as defined previously in (18), and ω_j' is the number of partitions "active" at row j of A, more precisely,

$$\omega_j' \overset{def}{=} |\{l\in[c]:\{i\in\mathcal{P}_l : A_{ji}\neq 0\}\neq\emptyset\}|, \qquad j\in[d]. \tag{27}$$

Proof When $m=1$, the result is equivalent to Theorem 4.1 in [5]. The extension to blocks ($m>1$) is straightforward. ■

Lemma 6 is a special case of Lemma 8 when only a single node ($c=1$) is used, in which case $\omega_j'=1$ for all $j\in[d]$. Lemma 8 also improves the constants $\{v_i\}$ derived in [21], where instead of ω_j and ω_j' in (26) one has $\max_j\omega_j$ and $\max_j\omega_j'$.

Lemma 8 is expressed in terms of certain sparsity parameters associated with the data ($\{\omega_j\}$) and the partitioning ($\{\omega_j'\}$). However, it is possible to derive alternative ESO results for the (c,τ)-distributed sampling. For instance, one can instead express the parameters $\{v_j\}$ without any sparsity assumptions, using only spectral properties of the data only. We have not included these results here, but in the $m=1$ case such results have been derived in [5]. It is possible to adopt them to the $m=1$ case as we have done it with Lemma 8.

4 Main Result

The complexity of our method is given by the following theorem.

Theorem 9 (Main Result) *Let Assumption 2 (ϕ_i are $(1/\gamma)$-smooth) and Assumption 3 (g is 1-strongly convex) be satisfied. Let $\hat{S}$ be a proper sampling (Assumption 1) and $v_1,\dots,v_n$ be positive scalars satisfying Assumption 4. Then the sequence of primal and dual variables $\{w^t,\alpha^t\}_{t\geq 0}$ of Quartz (Algorithm 1) satisfies:*

$$\mathbb{E}[P(w^t)-D(\alpha^t)]\leq(1-\theta)^t(P(w^0)-D(\alpha^0)), \tag{28}$$

where

$$\theta=\min_i\frac{p_i\lambda\gamma n}{v_i+\lambda\gamma n}. \tag{29}$$

In particular, if we fix $\epsilon \leq P(w^0) - D(\alpha^0)$*, then for*

$$T \geq \max_i \left(\frac{1}{p_i} + \frac{v_i}{p_i \lambda\gamma n}\right) \log\left(\frac{P(w^0) - D(\alpha^0)}{\epsilon}\right), \tag{30}$$

we are guaranteed that $\mathbb{E}[P(w^T) - D(\alpha^T)] \leq \epsilon$*.*

A result of a similar flavour but for a different problem and not in a primal-dual setting has been established in [22], where the authors analyze a parallel coordinate descent method, NSync, also with an *arbitrary sampling*, for minimizing a strongly convex function under an ESO assumption.

In the rest of this section we will specialize the above result to a few selected samplings. We then devote two separate sections to Quartz specialized to the τ-nice sampling (Section 5) and Quartz specialized to the (c, τ)-distributed sampling (Section 6 – as we do a more detailed analysis of the results in these two cases.

4.1 Quartz with uniform serial sampling

We first look at the special case when $\hat{S}$ is the uniform serial sampling, i.e., when $p_i = 1/n$ for all $i \in [n]$.

Corollary 10 *Assume that at each iteration of Quartz we update only one dual variable uniformly at random and use* $v_i = \lambda_{\max}(A_i^\top A_i)$ *for all* $i \in [n]$*. If we let* $\epsilon \leq P(w^0) - D(\alpha^0)$ *and*

$$T \geq \left(n + \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma}\right) \log\left(\frac{P(w^0) - D(\alpha^0)}{\epsilon}\right), \tag{31}$$

then $\mathbb{E}[P(w^T) - D(\alpha^T)] \leq \epsilon$*.*

Proof The result follows by combining Lemma 5 and Theorem 9. ■

Corollary 10 should be compared with Theorem 5 in [29] (covering the L2-regularized case) and Theorem 1 in [28] (covering the case of general g). They obtain the rate

$$\left(n + \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma}\right) \log\left(\left(n + \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma}\right)\left(\frac{D(\alpha^*) - D(\alpha^0)}{\epsilon}\right)\right),$$

where α^* is the dual optimal solution. Notice that the dominant terms in the two rates exactly match, although our logarithmic term is better and simpler.

4.2 Quartz with optimal serial sampling (importance sampling)

According to Lemma 5, the parameter v for a serial sampling $\hat{S}$ is determined by (16) and is *independent* of the distribution of $\hat{S}$. We can then seek to maximize the quantity θ in (29) to obtain the best bound. A simple calculation reveals that the optimal probability is given by:

$$\mathbb{P}(\hat{S} = \{i\}) = p_i^* \stackrel{\text{def}}{=} \frac{\lambda_{\max}(A_i^\top A_i) + \lambda\gamma n}{\sum_{i=1}^n \left(\lambda_{\max}(A_i^\top A_i) + \lambda\gamma n\right)}. \tag{32}$$

Using this sampling, we obtain the following iteration complexity bound, which is an improvement on the bound for uniform probabilities (31).

Corollary 11 *Assume that at each iteration of Quartz we update only one dual variable at random according to the probability p^* defined in (32) and use $v_i = \lambda_{\max}(A_i^\top A_i)$ for all $i \in [n]$. If we let $\epsilon \leq P(w^0) - D(\alpha^0)$ and*

$$T \geq \left(n + \frac{\frac{1}{n}\sum_{i=1}^n \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma} \right) \log\left(\frac{P(w^0) - D(\alpha^0)}{\epsilon} \right), \tag{33}$$

then $\mathbb{E}[P(w^T) - D(\alpha^T)] \leq \epsilon$.

Note that in contrast with the serial uniform sampling, we now have dependence on the *average* of the eigenvalues. The above result should be compared with the complexity result of Iprox-SDCA [37]:

$$\left(n + \frac{\frac{1}{n}\sum_{i=1}^n \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma} \right) \log\left(\left(n + \frac{\frac{1}{n}\sum_{i=1}^n \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma} \right) \left(\frac{D(\alpha^*) - D(\alpha^0)}{\epsilon} \right) \right),$$

where α^* is the dual optimal solution. Again, the dominant terms in the two rates exactly match, although our logarithmic term is better and simpler.

4.3 Quartz with product sampling

In this section we apply Theorem 9 to the case when $\hat{S}$ is the product sampling (see the description in Section 3.3). All the notation we use here was established there.

Corollary 12 *Under the group separability assumption, let $\hat{S}$ be the product sampling and let $v_i = \lambda_{\max}(A_i^\top A_i)$ for all $i \in [n]$. If we fix $\epsilon \leq P(w^0) - D(\alpha^0)$ and*

$$T \geq \max_i \left(|X_{l_i}| + \frac{\lambda_{\max}(A_i^\top A_i)|X_{l_i}|}{\lambda\gamma n} \right) \log\left(\frac{P(w^0) - D(\alpha^0)}{\epsilon} \right),$$

then $\mathbb{E}[P(w^T) - D(\alpha^T)] \leq \epsilon$.

Proof The proof follows directly from Theorem 9, Lemma 7 and (20). ■

Recall from Section 3.3 that the product sampling $\hat{S}$ has cardinality $\tau \geq 1$ and is non-uniform as long as all the sets $\{X_1, \ldots, X_\tau\}$ do not have the same cardinality. To the best of our knowledge, Corollary 12 is the first *explicit* complexity bound of stochastic algorithm using non-serial and non-uniform sampling for composite convex optimization problem (the paper [22] only deals with smooth functions and the method is not primal-dual), albeit under the group separability assumption.

Let us compare the complexity bound with the serial uniform case (Corollary 10):

$$\frac{n + \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma}}{\max_i \left(|X_{l_i}| + \frac{\lambda_{\max}(A_i^\top A_i)|X_{l_i}|}{\lambda\gamma n} \right)} \geq \min_i \frac{n}{|X_{l_i}|}.$$

Hence the iteration bound of Quartz specialized to product sampling is at most a $\max_i |X_{l_i}|/n$ fraction of that of Quartz specialized to serial uniform sampling. The factor $\max_i |X_{l_i}|/n$ varies from $1/\tau$ to 1, depending on the degree to which the partition $X_1, \ldots, X_\tau$ is balanced. A perfect

linear speedup ($\max_i |X_{l_i}|/n = 1/\tau$) only occurs when the partition $X_1, \ldots, X_\tau$ is perfectly balanced (i.e., the set X_l have the same cardinality), in which case the product sampling is uniform (recall the definition of uniformity we use in this paper: $\mathbb{P}(i \in \hat{S}) = \mathbb{P}(i' \in \hat{S})$ for all $i, i' \in [n]$). Note that if the partition is not perfectly but sufficiently so, then the factor $\max_i |X_{l_i}|/n$ will be close to the perfect linear speedup factor $1/\tau$.

5 Quartz with τ-nice Sampling (standard mini-batching)

We now specialize Theorem 9 to the case of the τ-nice sampling.

Corollary 13 *Assume $\hat{S}$ is the τ-nice sampling and v is chosen as in* (17)*. If we let $\epsilon \leq P(w^0) - D(\alpha^0)$ and*

$$T \geq \left(\frac{n}{\tau} + \frac{\max_i \lambda_{\max}\left(\sum_{j=1}^d \left(1 + \frac{(\omega_j - 1)(\tau - 1)}{n-1}\right) A_{ji}^\top A_{ji} \right)}{\lambda \gamma \tau} \right) \log \left(\frac{P(w^0) - D(\alpha^0)}{\epsilon} \right), \qquad (34)$$

then $\mathbb{E}[P(w^T) - D(\alpha^T)] \leq \epsilon$.

Proof The result follows by combining Lemma 6 and Theorem 9. ■

Let us now have a detailed look at the above result; especially in terms of how it compares with the serial uniform case (Corollary 10). We do this comparison in Table 1. For fully sparse data, we get *perfect linear speedup*: the bound in the second line of Table 1 is a $1/\tau$ fraction of the bound in the first line. For fully dense data, the condition number ($\kappa \overset{\text{def}}{=} \max_i \lambda_{\max}(A_i^\top A_i)/(\gamma\lambda)$) is unaffected by mini-batching/parallelization. Hence, linear speedup is obtained if $\kappa = O(n/\tau)$. For general data, the behaviour of Quartz with τ-nice sampling interpolates these two extreme cases. That is, κ gets multiplied by a quantity between $1/\tau$ (fully sparse case) and 1 (fully dense case). It is convenient to write this factor in the form

$$\frac{1}{\tau}\left(1 + \frac{(\tilde{\omega} - 1)(\tau - 1)}{n - 1}\right),$$

where $\tilde{\omega} \in [1, n]$ is a measure of *average sparsity* of the data, using which we can write:

$$T(\tau) \overset{\text{def}}{=} \left(\frac{n}{\tau} + \frac{\left(1 + \frac{(\tilde{\omega}-1)(\tau-1)}{n-1}\right) \max_i \lambda_{\max}(A_i^\top A_i)}{\lambda \gamma \tau} \right) \log \left(\frac{P(w^0) - D(\alpha^0)}{\epsilon} \right). \qquad (35)$$

5.1 Theoretical speedup factor

For simplicity of exposition, let us now assume that $\lambda_{\max}(A_i^\top A_i) = 1$. We will now study the *theoretical speedup factor*, defined as:

$$\frac{T(1)}{T(\tau)} \overset{(35)}{=} \frac{\tau(1 + \lambda\gamma n)}{1 + \lambda\gamma n + \frac{(\tau-1)(\tilde{\omega}-1)}{(n-1)}} = \frac{\tau}{1 + \frac{(\tau-1)(\tilde{\omega}-1)}{(n-1)(1+\lambda\gamma n)}} \quad . \qquad (36)$$

Sampling $\hat{S}$	Data	Complexity of Quartz (34)	Theorem
Serial uniform	Any data	$n + \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma}$	Corollary 10
τ-nice	Fully sparse data ($\omega_j = 1$ for all j)	$\frac{n}{\tau} + \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma\tau}$	Corollary 13
τ-nice	Fully dense data ($\omega_j = n$ for all j)	$\frac{n}{\tau} + \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma}$	Corollary 13
τ-nice	Any data	$\frac{n}{\tau} + \frac{\left(1 + \frac{(\tilde{\omega}-1)(\tau-1)}{n-1}\right) \max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma\tau}$	Corollary 13

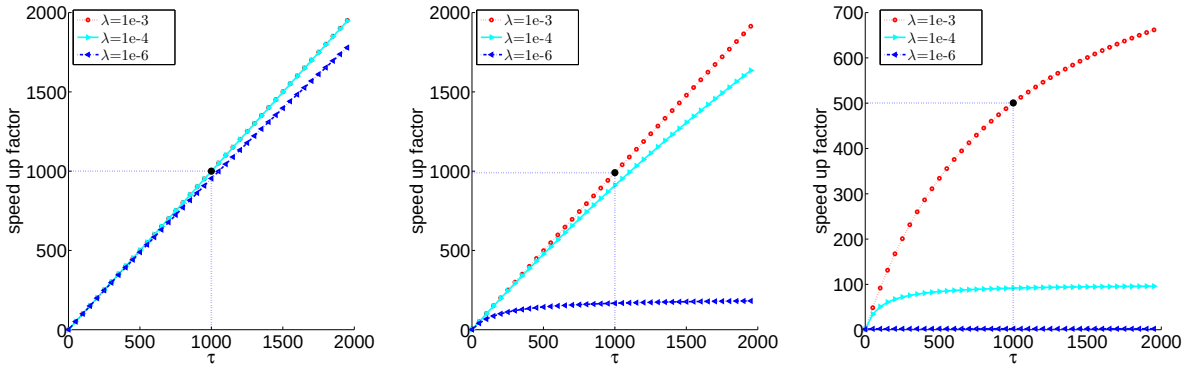
Table 1: Comparison of the complexity of Quartz with serial uniform sampling and τ-nice sampling.

That is, the speedup factor measures how much better Quartz is with τ-nice sampling than in the serial uniform case (with 1-nice sampling). Note that the speedup factor is a concave and increasing function with respect to the number of threads τ. The value depends on two factors: the *relative sparsity* level of the data matrix A, expressed through the quantity $\frac{\tilde{\omega}-1}{n-1}$ and the condition number of the problem, expressed through the quantity $\lambda\gamma n$. We provide below two lower bounds for the speedup factor:

$$\frac{T(1,1)}{T(1,\tau)} \geq \frac{\tau}{1 + \frac{\tilde{\omega}-1}{n-1}} \geq \frac{\tau}{2} \qquad \text{if} \qquad 1 \leq \tau \leq 2 + \lambda\gamma n \ . \tag{37}$$

Note that the last term does not involve $\tilde{\omega}$. In other words, linear speedup (modulus a factor of 2) is achieved at least until $\tau = 2 + \lambda\gamma n$ (of course, we also require that $\tau \leq n$), regardless of the data matrix A. For instance, if $\lambda\gamma = 1/\sqrt{n}$, which is a frequently used setting for the regularizer, then we get *data independent linear speedup* up to mini-batch size $\tau = 2 + \sqrt{n}$. Moreover, from the first inequality in (37) we see that there is *further data-dependent speedup*, depending on the average sparsity measure $\tilde{\omega}$. We give an illustration of this phenomenon in Figure 1, where we plot the theoretical speedup factor (36) as a function of the number of threads τ, for $n = 10^6$, $\gamma = 1$ and three values of $\tilde{\omega}$ and λ. Lookingat the plots from right to left, we see that for fixed λ, the speedup factor increases as $\tilde{\omega}$ decreases, as described by (36). Moreover, as the regularization parameter λ gets smaller an reaches the value $1/n$, the speedup factor is healthy for sparse data only. However, for $\lambda = 1/\sqrt{n} = 10^{-3}$, we observe linear speedup up to $\tau = \sqrt{n} = 1000$, regardless

of $\tilde{\omega}$ (the sparsity of the data), as predicted. There is additional data-driven speedup beyond this point, which is better for sparser data.



(a) $\tilde{\omega} = 10^2$, $n = 10^6$, $\gamma = 1$ (b) $\tilde{\omega} = 10^4$, $n = 10^6$, $\gamma = 1$ (c) $\tilde{\omega} = 10^6$, $n = 10^6$, $\gamma = 1$

Figure 1: The speedup factor (36) as a function of τ for $n = 10^6$, $\gamma = 1$, three regularization parameters and data of various sparsity levels.

5.2 Quartz vs existing primal-dual mini-batch methods

We now compare the above result with existing mini-batch stochastic dual coordinate ascent methods. A mini-batch variant of SDCA, to which Quartz with τ-nice sampling can be naturally compared, has been proposed and analyzed previously in [30], [28] and [36]. In [30], the authors proposed to use a so-called *safe mini-batching*, which is precisely equivalent to finding the stepsize parameter v satisfying Assumption 4 (in the special case of τ-nice sampling). However, they only analyzed the case where the functions ϕ_i are non-smooth. In [28], the authors studied accelerated mini-batch SDCA (ASDCA), specialized to the case when the regularizer g is the squared L2 norm. They showed that the complexity of ASDCA interpolates between that of SDCA and accelerated gradient descent (AGD) [20] through varying the mini-batch size τ. In [36], the authors proposed a mini-batch extension of their stochastic primal-dual coordinate algorithm (SPDC). Both ASDCA and SPDC reach the same complexity as AGD when the mini-batch size equals to n, thus should be considered as accelerated algorithms. The complexity bounds for all these algorithms are summarized in Table 2. To facilitate the comparison, we assume that $\max_i \lambda_{\max}(A_i^\top A_i) = 1$ (since the analysis of ASDCA assumes this). In Table 3 we compare the complexities of SDCA, ASDCA, SPDC and Quartz in several regimes. We have used Lemma 14 to simplify the bounds for Quartz.

Lemma 14 *For any $\tilde{\omega} \in [1, n]$ and $\tau \in [1, n]$ we have*

$$\frac{(\tilde{\omega}-1)(\tau-1)}{n-1} \leq \frac{\tilde{\omega}\tau}{n} \leq 1 + \frac{(\tilde{\omega}-1)(\tau-1)}{n-1} \leq 1 + \frac{\tilde{\omega}\tau}{n}.$$

Proof The second inequality follows by showing that the function $\phi_1(x) = x + \frac{(\tilde{\omega}-x)(\tau-x)}{n-x}$ is increasing, the first and third follow by showing that $\phi_2(x) = \frac{(\tilde{\omega}-x)(\tau-x)}{n-x}$ is decreasing on $[0, 1]$. The monotonicity claims follow from the fact that $\phi_1'(x) = \frac{n^2+\tilde{\omega}\tau-n(\tilde{\omega}+\tau)}{(n-x)^2} = \frac{(n-\tilde{\omega})(n-\tau)}{(n-x)^2} \geq 0$ and

Algorithm	Iteration complexity	g
SDCA [29]	$n+\frac{1}{\lambda\gamma}$	$\frac{1}{2}\|\cdot\|^2$
ASDCA [28]	$4\times\max\left\{\frac{n}{\tau},\sqrt{\frac{n}{\lambda\gamma\tau}},\frac{1}{\lambda\gamma\tau},\frac{n^{\frac{1}{3}}}{(\lambda\gamma\tau)^{\frac{2}{3}}}\right\}$	$\frac{1}{2}\|\cdot\|^2$
SPDC [36]	$\frac{n}{\tau}+\sqrt{\frac{n}{\lambda\gamma\tau}}$	general
Quartz with τ-nice sampling	$\frac{n}{\tau}+\left(1+\frac{(\tilde{\omega}-1)(\tau-1)}{n-1}\right)\frac{1}{\lambda\gamma\tau}$	general

Table 2: Comparison of the iteration complexity of several primal-dual algorithms performing stochastic coordinate ascent steps in the dual using a mini-batch of examples of size τ (with the exception of SDCA, which is a serial method using $\tau=1$. We assume that $\lambda_{\max}(A_i^\top A_i)=1$ for all i to facilitate comparison since this assumption has been implictly made in [28].

Algorithm	$\gamma\lambda n=\Theta(\frac{1}{\sqrt{n}})$	$\gamma\lambda n=\Theta(\frac{1}{\tau})$	$\gamma\lambda n=\Theta(1)$	$\gamma\lambda n=\Theta(\tau)$	$\gamma\lambda n=\Theta(\sqrt{n})$
	$\kappa=n^{3/2}$	$\kappa=n\tau$	$\kappa=n$	$\kappa=n/\tau$	$\kappa=\sqrt{n}$
SDCA [29]	$n^{3/2}$	$n\tau$	n	n	n
ASDCA [28]	$n^{3/2}/\tau+$ $n^{5/4}/\sqrt{\tau}+$ $n^{4/3}/\tau^{2/3}$	n	$n/\sqrt{\tau}$	n/τ	$n/\tau+n^{3/4}/\sqrt{\tau}$
SPDC [36]	$n^{5/4}/\sqrt{\tau}$	n	$n/\sqrt{\tau}$	n/τ	$n/\tau+n^{3/4}/\sqrt{\tau}$
Quartz (τ-nice)	$n^{3/2}/\tau+\tilde{\omega}\sqrt{n}$	$n+\tilde{\omega}\tau$	$n/\tau+\tilde{\omega}$	n/τ	$n/\tau+\tilde{\omega}/\sqrt{n}$

Table 3: Comparison of leading factors in the complexity bounds of several methods in 5 regimes; where $\kappa=1/(\gamma\lambda)$ is the condition number. We ignore constant terms and hence one can replace each "plus" by a "max".

$\phi_2'(x)=\phi_1'(x)-1=\frac{(n-\tilde{\omega})(n-\tau)-(n-x)^2}{(n-x)^2}\leq 0$ for all $x\in[0,1]$. ■

Looking at Table 3, we see that in the $\gamma\lambda n=\Theta(\tau)$ regime (i.e., if the condition number is $\kappa=\Theta(n/\tau)$), Quartz matches the linear speedup (when compared to SDCA) of ASDCA and SPDC. When the condition number is roughly equal to the sample size ($\kappa=\Theta(n)$), then Quartz does better than both ASDCA and SPDC as long as $n/\tau+\tilde{\omega}\leq n/\sqrt{\tau}$. In particular, this is the case when the data is sparse: $\tilde{\omega}\leq n/\sqrt{\tau}$. If the data is even more sparse (and in many big data applications one has $\tilde{\omega}=O(1)$) and we have $\tilde{\omega}\leq n/\tau$, then Quartz significantly outperforms both ASDCA and SPDC. Note that Quartz can be better than both ASDCA and SPDC even in the domain of accelerated methods, that is, when the condition number is larger than the number of examples:

$$\kappa=\frac{1}{\gamma\lambda}\geq n. \tag{38}$$

Indeed, we have the following result, which can be interpreted as follows: if $\kappa\leq\tau n/4$ (that is, $\lambda\gamma\tau n\geq 4$), then there are sparse-enough problems for which Quartz is better than both ASDCA

and SPDC.

Proposition 15 *Assume that* (38) *holds and that* $\max_i \lambda_{max}(A_i^\top A_i) = 1$. *Then if the data is sufficiently sparse so that*

$$\lambda\gamma\tau n \geq \left(2 + \frac{(\tilde{\omega}-1)(\tau-1)}{n-1}\right)^2, \tag{39}$$

the iteration complexity (in $\tilde{O}$ *order) of Quartz is better than that of ASDCA and SPDC.*

Proof As long as $\lambda\gamma\tau n \geq 1$, which holds under our assumption, the iteration complexity of ASDCA is:

$$\tilde{O}\left(\max\left\{\frac{n}{\tau}, \sqrt{\frac{n}{\lambda\gamma\tau}}, \frac{1}{\lambda\gamma\tau}, \frac{n^{\frac{1}{3}}}{(\lambda\gamma\tau)^{\frac{2}{3}}}\right\}\right) = \tilde{O}\left(\sqrt{\frac{n}{\lambda\gamma\tau}}\right).$$

which is already less than that of SPDC. Moreover,

$$\sqrt{\frac{n}{\lambda\gamma\tau}} \overset{(39)}{\geq} \frac{2 + \frac{(\tau-1)(\tilde{\omega}-1)}{n-1}}{\lambda\gamma\tau} \overset{(38)}{\geq} \frac{n}{\tau} + \frac{1 + \frac{(\tau-1)(\tilde{\omega}-1)}{n-1}}{\lambda\gamma\tau}.$$

■

6 Quartz with Distributed Sampling

In this section we apply Theorem 9 to the case when $\hat{S}$ is the (c,τ)-distributed sampling; see the description of this sampling in Section 3.4.

Corollary 16 *Assume that* $\hat{S}$ *is a* (c,τ)*-distributed sampling and* v *is chosen as in* (26). *If we let* $\epsilon \leq P(w^0) - D(\alpha^0)$ *and*

$$T \geq T(c,\tau) \times \log\left(\frac{P(w^0) - D(\alpha^0)}{\epsilon}\right), \tag{40}$$

where

$$T(c,\tau) \overset{def}{=} \frac{n}{c\tau} + \max_i \frac{\lambda_{\max}\left(\sum_{j=1}^d \left(1 + \frac{(\tau-1)(\omega_j-1)}{\max\{n/c-1,1\}} + \left(\frac{\tau c}{n} - \frac{\tau-1}{\max\{n/c-1,1\}}\right)\frac{\omega_j'-1}{\omega_j'}\omega_j\right) A_{ji}^\top A_{ji}\right)}{\lambda\gamma c\tau}, \tag{41}$$

then $\mathbb{E}[P(w^T) - D(\alpha^T)] \leq \epsilon$.

Proof If $\hat{S}$ is a (c,τ)-distributed sampling, then

$$p_i = \frac{c\tau}{n}, \qquad i \in [n].$$

It now only remains to combine Theorem 9 and Lemma 8. ■

The expression (41) involves ω_j', which depends on the partitioning $\{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_c\}$ of the dual variable and the data. The following lemma says that the effect of the partition is negligible, and in fact vanishes as τ increases. It was proved in [5, Lemma 5.2].

Lemma 17 ([5]) *If $n/c \geq 2$ and $\tau \geq 2$, then for all $j \in [d]$, we have*

$$\left(\frac{\tau c}{n} - \frac{\tau - 1}{n/c - 1}\right) \frac{\omega_j' - 1}{\omega_j'} \omega_j \leq \frac{1}{\tau - 1}\left(1 + \frac{(\tau - 1)(\omega_j - 1)}{n/c - 1}\right).$$

According to this result, when each node owns at least two dual examples ($n/c \geq 2$) and picks and updates at least two examples in each iteration ($\tau \geq 2$), then

$$\begin{aligned} T(c,\tau) &\leq \frac{n}{c\tau} + \left(1 + \frac{1}{\tau - 1}\right) \frac{\max_i \lambda_{\max}\left(\sum_{j=1}^d \left(1 + \frac{(\tau-1)(\omega_j - 1)}{n/c - 1}\right) A_{ji}^\top A_{ji}\right)}{\lambda\gamma c\tau} \\ &= \frac{n}{c\tau} + \left(1 + \frac{1}{\tau - 1}\right)\left(1 + \frac{(\tau - 1)(\hat{\omega} - 1)}{n/c - 1}\right) \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma c\tau}, \end{aligned} \tag{42}$$

where $\hat{\omega} \in [1, n]$ is an *average sparsity measure* similar to that one we introduced in the study of τ-nice sampling. This bound is similar to that we obtained for the τ-nice sampling; and can be interpreted in an analogous way. Note that as the first term (n) receives perfect mini-batch scaling (it is divided by $c\tau$), while the condition number $\max_i \lambda_{\max}(A_i^\top A_i)/(\lambda\gamma)$ is divided by $c\tau$ but also multiplied by $\left(1 + \frac{1}{\tau - 1}\right)\left(1 + \frac{(\tau-1)(\hat{\omega}-1)}{n/c-1}\right)$. However, this term is bounded by $2\hat{\omega}$, and hence if $\hat{\omega}$ is small, the condition number also receives a nearly perfect mini-batch scaling.

6.1 Quartz vs DiSDCA

A distributed variant of SDCA, named DisDCA, has been proposed in [34] and analyzed in [35]. The authors of [34] proposed a basic DisDCA variant (which was analyzed) and a practical DisDCA variant (which was not analyzed). The complexity of basic DisDCA was shown to be:

$$\left(\frac{n}{c\tau} + \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma}\right) \log\left(\frac{n}{c\tau} + \left(\frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma}\right) \cdot \frac{D(\alpha^*) - D(\alpha^0)}{\epsilon}\right), \tag{43}$$

where α^* is an optimal dual solution. Note that this rate is much worse than our rate. Ignoring the logarithmic terms, while the first expression $n/(c\tau)$ is the same in both results, if we replace all ω_j by the upper bound n and all ω_j' by the upper bound c in (41), then

$$\begin{aligned} T(c,\tau) &\leq \frac{n}{c\tau} + \left(\max_i \lambda_{\max}(A_i^\top A_i)\right) \cdot \frac{1 + \frac{(\tau-1)(n-1)}{\max(n/c-1)} + (\frac{\tau c}{n} - \frac{\tau - 1}{\max(n/c-1,1)})\frac{c-1}{c} n}{\lambda\gamma c\tau} \\ &\leq \frac{n}{c\tau} + \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma}. \end{aligned}$$

Therefore, the dominant term in (40) is a strict lower bound of that in (43). Moreover, it is clear that the gap between (40) and (43) is large when the data is sparse. For instance, in the perfectly sparse case with $\hat{\omega} = 1$, the bound (42) for Quartz becomes

$$\frac{n}{c\tau} + \left(1 + \frac{1}{\tau - 1}\right) \frac{\max_i \lambda_{\max}(A_i^\top A_i)}{\lambda\gamma c\tau},$$

which is much better than (43).

6.2 Theoretical speedup factor

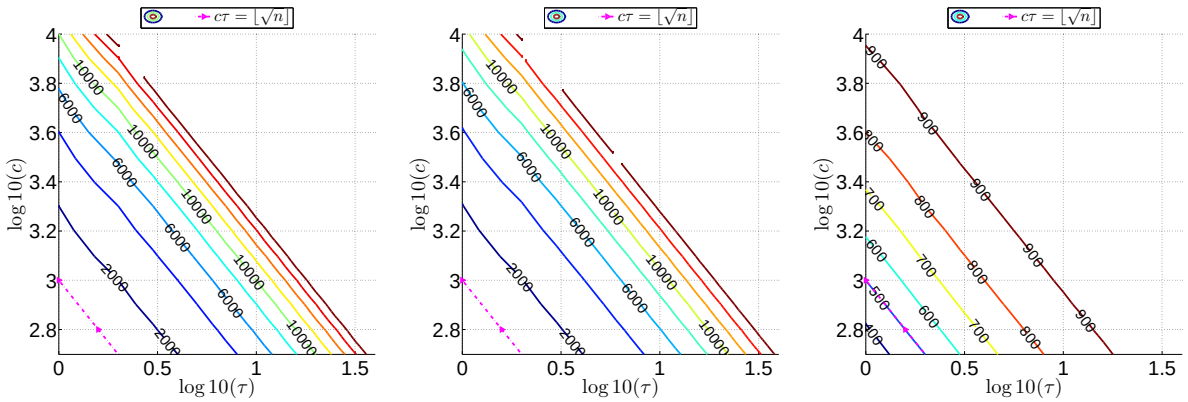
In analogy with the discussion in Section 5.1, we shall now analyze the theoretical speedup factor $T(1,1)/T(c,\tau)$ measuring the multiplicative amount by which Quartz specialized to the (c,τ)-distributed sampling is better than Quartz specialized to the serial uniform sampling.

In Section 5, we have seen how the speedup factor increases with τ when a mini-batch of examples is used at each iteration following the τ-nice sampling. As we have discussed before, this sampling is not particularly suitable for a distributed implementation (unless $\tau = n$; which in the big data setting where n is very large may be asking for many more cores/threads that are available). This is because the implementation of updates using this sampling would either result in frequently idle nodes, or in increased data transfer.

Often the data matrix A is too large to be stored on a single node, or limited number of threads/cores are available per node. We then want to implement Quartz in a distributed way ($c > 1$). It is therefore necessary to understand how the speedup factor compares to the hypothetical situation in which we would have a large machine where all data could be stored (we ignore communication costs here) and hence a $c\tau$-nice sampling could be implemented. That is, we are interested in comparing $T(c,\tau)$ (distributed implementation) and $T(1,c\tau)$ (hypothetical computer). If for simplicity of exposition we assume that $\lambda_{\max}(A_i^\top A_i) = 1$, it is possible to argue that if $c\tau \leq n$, then

$$\frac{T(1,1)}{T(c,\tau)} \approx \frac{T(1,1)}{T(1,c\tau)}. \tag{44}$$

In Figure 2 we plot the contour lines of the theoretical speedup factor in a log-log plot with axes corresponding to τ and c. The contours are nearly perfect straight lines, which means that the speedup factor is approximately constant for those pairs (c,τ) for which $c\tau$ is the same. In particular, this means that (44) holds. Note that better speedup is obtained for sparse data then for dense data. However, in all plots we have chosen $\gamma = 1$ and $\lambda = 1/\sqrt{n}$; and hence we expect data independent linear speedup up to $c\tau = \Theta(\sqrt{n})$ – a special line is depicted in all three plots which defines this contour.



(a) $\omega = 10^2$, $n = 10^6$ (b) $\omega = 10^4$, $n = 10^6$ (c) $\omega = 10^6$, $n = 10^6$

Figure 2: Contour line plots of the speedup factor $T(1,1)/T(c,\tau)$ for $n = 10^6$, $\gamma = 1$, $\lambda = 10^{-3}$, $\omega = 10^2$ (Figure 2(a)), $\omega = 10^4$ (Figure 2(b)), $\omega = 10^6$ (Figure 2(c)). Here, $\omega \in [1,n]$ is a degree of average sparsity of the data.

7 Proof of the Main Result

In this section we prove our main result (Theorem 9). In order to make the analysis more transparent, we will first establish three auxiliary results.

7.1 Three lemmas

Lemma 18 *Function $f:\mathbb{R}^N\to\mathbb{R}$ defined in (3) satisfies the following inequality:*

$$f(\alpha+h)\leq f(\alpha)+\langle\nabla f(\alpha),h\rangle+\frac{1}{2\lambda n^2}h^\top A^\top Ah,\qquad\forall\alpha,h\in\mathbb{R}^N. \tag{45}$$

Proof Since g is 1-strongly convex, g^* is 1-smooth. Pick $\alpha,h\in\mathbb{R}^N$. Since, $f(\alpha)=\lambda g^*(\frac{1}{\lambda n}A\alpha)$, we have

$$\begin{aligned}f(\alpha+h)=\lambda g^*\left(\tfrac{1}{\lambda n}A\alpha+\tfrac{1}{\lambda n}Ah\right) &\leq \lambda\left(g^*\left(\tfrac{1}{\lambda n}A\alpha\right)+\langle\nabla g^*\left(\tfrac{1}{\lambda n}A\alpha\right),\tfrac{1}{\lambda n}Ah\rangle+\tfrac{1}{2}\left\|\tfrac{1}{\lambda n}Ah\right\|^2\right)\\ &= f(\alpha)+\langle\nabla f(\alpha),h\rangle+\tfrac{1}{2\lambda n^2}h^TA^\top Ah.\end{aligned}$$

■

For $s=(s_1,\dots,s_n)\in\mathbb{R}^N$, $h=(h_1,\dots,h_n)\in R^N$, where $s_i,h_i\in\mathbb{R}^m$ for all i, we will for convenience write

$$\langle s,h\rangle_p=\sum_{i=1}^n p_i\langle s_i,h_i\rangle,$$

where $p=(p_1,\dots,p_n)$ and $p_i=\mathbb{P}(i\in\hat{S})$ for $i\in[n]$.

In the next lemma we give an expected separable overapproximation of the convex function $-D$.

Lemma 19 *If $\hat{S}$ and $v\in\mathbb{R}^n$ satisfy Assumption 4, then for all $\alpha,h\in\mathbb{R}^N$, the following holds:*

$$\begin{aligned}&\mathbb{E}[-D(\alpha+h_{[\hat{S}]})]\\ &\leq f(\alpha)+\langle\nabla f(\alpha),h\rangle_p+\frac{1}{2\lambda n^2}\|h\|^2_{p\cdot v}+\frac{1}{n}\sum_{i=1}^n\left[(1-p_i)\phi_i^*(-\alpha_i)+p_i\phi_i^*(-\alpha_i-h_i)\right].\end{aligned} \tag{46}$$

Proof By definition of D, we have

$$-D(\alpha+h_{[\hat{S}]})\overset{(2)}{=}f(\alpha+h_{[\hat{S}]})+\psi(\alpha+h_{[\hat{S}]}),$$

where f and ψ are defined in (3) and (4). Now we apply Lemma 18 and (15) to bound the first term:

$$\begin{aligned}\mathbb{E}[f(\alpha+h_{[\hat{S}]})]&\overset{(45)}{\leq}\mathbb{E}[f(\alpha)+\langle\nabla f(\alpha),h_{[\hat{S}]}\rangle+\frac{1}{2\lambda n^2}h_{[\hat{S}]}^\top A^\top Ah_{[\hat{S}]}]\\ &\overset{(15)}{\leq} f(\alpha)+\mathbb{E}[\langle\nabla f(\alpha),h_{[\hat{S}]}\rangle]+\frac{1}{2\lambda n^2}\|h\|^2_{p\cdot v}\\ &= f(\alpha)+\langle\nabla f(\alpha),h\rangle_p+\frac{1}{2\lambda n^2}\|h\|^2_{p\cdot v}.\end{aligned}$$

Moreover, since ψ is block separable, we can write

$$\begin{aligned}\mathbb{E}[\psi(\alpha+h_{[\hat{S}]})] &\overset{(4)}{=} \frac{1}{n}\sum_{i=1}^{n}\left[\mathbb{P}(i\notin\hat{S})\phi_i^*(-\alpha_i)+\mathbb{P}(i\in\hat{S})\phi_i^*(-\alpha_i-h_i)\right]\\ &= \frac{1}{n}\sum_{i=1}^{n}\left[(1-p_i)\phi_i^*(-\alpha_i)+p_i\phi_i^*(-\alpha_i-h_i)\right].\end{aligned}$$

■

Our last auxiliary result is a technical lemma for further bounding the right hand side in Lemma 19.

Lemma 20 *Suppose that $\hat{S}$ and $v\in\mathbb{R}^n$ satisfy Assumption 4. Fixing $\alpha\in\mathbb{R}^N$ and $w\in\mathbb{R}^d$, let $h\in\mathbb{R}^N$ be defined by:*

$$h_i=-\theta p_i^{-1}(\alpha_i+\nabla\phi_i(A_i^\top w)),\ \ i\in[n],$$

where θ be as in (29)*. Then*

$$\begin{aligned}&f(\alpha)+\langle\nabla f(\alpha),h\rangle_p+\frac{1}{2\lambda n^2}\|h\|_{p\cdot v}^2+\frac{1}{n}\sum_{i=1}^{n}\left[(1-p_i)\phi_i^*(-\alpha_i)+p_i\phi_i^*(-\alpha_i-h_i)\right]\\ &\le -(1-\theta)D(\alpha)-\theta\lambda g(\nabla g^*(\bar{\alpha}))-\frac{1}{n}\sum_{i=1}^{n}\langle\theta\nabla g^*(\bar{\alpha}),A_i\nabla\phi_i(A_i^\top w)\rangle+\frac{\theta}{n}\sum_{i=1}^{n}\phi_i^*(\nabla\phi_i(A_i^\top w)),\end{aligned}\tag{47}$$

where $\bar{\alpha}=\frac{1}{\lambda n}A\alpha$.

Proof Recall from (3) that $f(\alpha)=\lambda g^*(\bar{\alpha})$ and hence $\nabla f(\alpha)=\frac{1}{n}A^\top\nabla g^*(\bar{\alpha})$. Thus,

$$\begin{aligned}&f(\alpha)+\langle\nabla f(\alpha),h\rangle_p+\frac{1}{2\lambda n^2}\|h\|_{p\cdot v}^2\\ &=\lambda g^*(\bar{\alpha})-\sum_{i=1}^{n}p_i\langle\frac{1}{n}A_i^\top\nabla g^*(\bar{\alpha}),\theta p_i^{-1}(\alpha_i+\nabla\phi_i(A_i^\top w))\rangle+\frac{1}{2\lambda n^2}\|h\|_{p\cdot v}^2\\ &=(1-\theta)\lambda g^*(\bar{\alpha})+\theta\lambda(g^*(\bar{\alpha})-\langle\nabla g^*(\bar{\alpha}),\bar{\alpha}\rangle)\\ &\quad-\frac{1}{n}\sum_{i=1}^{n}\langle\theta\nabla g^*(\bar{\alpha}),A_i\nabla\phi_i(A_i^\top w)\rangle+\frac{1}{2\lambda n^2}\|h\|_{p\cdot v}^2.\end{aligned}\tag{48}$$

Since the functions ϕ_i are $(1/\gamma)$-smooth, the conjugate functions ϕ_i^* must be γ-strongly convex. Therefore,

$$\begin{aligned}&\phi_i^*(-\alpha_i-h_i)\\ &=\phi_i^*(-(1-\theta p_i^{-1})\alpha_i+\theta p_i^{-1}\nabla\phi_i(A_i^\top w))\\ &\le(1-\theta p_i^{-1})\phi_i^*(-\alpha_i)+\theta p_i^{-1}\phi_i^*(\nabla\phi_i(A_i^\top w))-\frac{\gamma\theta p_i^{-1}(1-\theta p_i^{-1})}{2}\|\alpha_i+\nabla\phi_i(A_i^\top w)\|^2\\ &=(1-\theta p_i^{-1})\phi_i^*(-\alpha_i)+\theta p_i^{-1}\phi_i^*(\nabla\phi_i(A_i^\top w))-\frac{\gamma p_i(1-\theta p_i^{-1})}{2\theta}\|h_i\|^2,\end{aligned}\tag{49}$$

and we can write

$$\begin{aligned}&\frac{1}{n}\sum_{i=1}^{n}[(1-p_i)\phi_i^*(-\alpha_i)+p_i\phi_i^*(-\alpha_i-h_i)]\\&\overset{(49)}{\leq}(1-\theta)\psi(\alpha)+\frac{\theta}{n}\sum_{i=1}^{n}(\phi_i^*(\nabla\phi_i(A_i^\top w)))-\frac{1}{2\lambda n^2}\sum_{i=1}^{n}\frac{n\lambda\gamma p_i^2(1-\theta p_i^{-1})}{\theta}\|h_i\|^2.\end{aligned}\tag{50}$$

Then by combining (48) and (50) we get:

$$\begin{aligned}&f(\alpha)+\langle\nabla f(\alpha),h\rangle_p+\frac{1}{2\lambda n^2}\|h\|_{p\cdot v}^2+\frac{1}{n}\sum_{i=1}^{n}[(1-p_i)\phi_i^*(-\alpha_i)+p_i\phi_i^*(-\alpha_i-h_i)]\\&\leq-(1-\theta)D(\alpha)-\theta\lambda g(\nabla g^*(\bar{\alpha}))-\frac{1}{n}\sum_{i=1}^{n}\langle\theta\nabla g^*(\bar{\alpha}),A_i\nabla\phi_i(A_i^\top w)\rangle+\frac{\theta}{n}\sum_{i=1}^{n}\phi_i^*(\nabla\phi_i(A_i^\top w))\\&\quad+\frac{1}{2\lambda n^2}\sum_{i=1}^{n}\left(p_iv_i-\frac{n\lambda\gamma p_i^2(1-\theta p_i^{-1})}{\theta}\right)\|h_i\|^2.\end{aligned}$$

It remains to notice that for θ defined in (29), we have:

$$p_iv_i\leq\frac{n\lambda\gamma p_i^2(1-\theta p_i^{-1})}{\theta},\ \ \forall i\in[n].$$

■

7.2 Proof of Theorem 9

Let $t\geq 1$. Define $h^t=(h_1^t,\ldots,h_n^t)\in\mathbb{R}^N$ by:

$$h_i^t=-\theta p_i^{-1}(\alpha_i^{t-1}+\nabla\phi_i(A_i^\top w^t)),\ \ i\in[n]$$

and $\kappa^t=(\kappa_1^t,\cdots,\kappa_n^t)$ by:

$$\kappa_i^t=\arg\max_{\Delta\in\mathbb{R}^m}\left[-\phi_i^*(-(\alpha_i^{t-1}+\Delta))-\nabla g^*(\bar{\alpha}^{t-1})^\top A_i\Delta-\frac{v_i\|\Delta\|^2}{2\lambda n}\right],\ \ \forall i\in[n].$$

If we use Option I in Algorithm 1, then $\alpha^t=\alpha^{t-1}+\kappa_{[\hat{S}]}^t$. If we use Option II in Algorithm 1, then we have $\alpha^t=\alpha^{t-1}+h_{[\hat{S}]}^t$. In both cases, by Lemma 19:

$$\begin{aligned}&\mathbb{E}_t[-D(\alpha^t)]\\&\leq f(\alpha^{t-1})+\langle\nabla f(\alpha^{t-1}),h^t\rangle_p+\frac{1}{2\lambda n^2}\|h^t\|_{p\cdot v}^2+\frac{1}{n}\sum_{i=1}^{n}\left[(1-p_i)\phi_i^*(-\alpha_i^{t-1})+p_i\phi_i^*(-\alpha_i^{t-1}-h_i^t)\right].\end{aligned}$$

We now apply Lemma 20 to further bound the last term and obtain:

$$\begin{aligned}\mathbb{E}_t[-D(\alpha^t)]\leq&-(1-\theta)D(\alpha^{t-1})-\theta\lambda g(\nabla g^*(\bar{\alpha}^{t-1}))\\&-\frac{1}{n}\sum_{i=1}^{n}\langle\theta\nabla g^*(\bar{\alpha}^{t-1}),A_i\nabla\phi_i(A_i^\top w^t)\rangle+\frac{\theta}{n}\sum_{i=1}^{n}\phi_i^*(\nabla\phi_i(A_i^\top w^t)).\end{aligned}\tag{51}$$

By convexity of g,

$$\begin{aligned} P(w^t) &= \frac{1}{n}\sum_{i=1}^n \phi_i(A_i^\top w^t) + \lambda g((1-\theta)w^{t-1} + \theta\nabla g^*(\bar{\alpha}^{t-1})) \\ &\leq \frac{1}{n}\sum_{i=1}^n \phi_i(A_i^\top w^t) + (1-\theta)\lambda g(w^{t-1}) + \theta\lambda g(\nabla g^*(\bar{\alpha}^{t-1})). \end{aligned} \tag{52}$$

By combining (51) and (52) we get:

$$\begin{aligned} \mathbb{E}_t[P(w^t) - D(\alpha^t)] \leq& \frac{1}{n}\sum_{i=1}^n \phi_i(A_i^\top w^t) + (1-\theta)\lambda g(w^{t-1}) - (1-\theta)D(\alpha^{t-1}) \\ &- \frac{1}{n}\sum_{i=1}^n \langle \theta\nabla g^*(\bar{\alpha}^{t-1}), A_i\nabla\phi_i(A_i^\top w^t)\rangle + \frac{\theta}{n}\sum_{i=1}^n \phi_i^*(\nabla\phi_i(A_i^\top w^t)) \\ =& (1-\theta)(P(w^{t-1}) - D(\alpha^{t-1})) + \frac{1}{n}\sum_{i=1}^n (\phi_i(A_i^\top w^t) - (1-\theta)\phi_i(A_i^\top w^{t-1})) \\ &- \frac{1}{n}\sum_{i=1}^n \langle \theta\nabla g^*(\bar{\alpha}^{t-1}), A_i\nabla\phi_i(A_i^\top w^t)\rangle + \frac{\theta}{n}\sum_{i=1}^n \phi_i^*(\nabla\phi_i(A_i^\top w^t)). \end{aligned} \tag{53}$$

Note that $\theta\nabla g^*(\bar{\alpha}^{t-1}) = w^t - (1-\theta)w^{t-1}$ and $\phi_i^*(\nabla\phi_i(A_i^\top w^t)) = \langle \nabla\phi_i(A_i^\top w^t), A_i^\top w^t\rangle - \phi_i(A_i^\top w^t)$. Finally, we plug these two inequalities into (53) and obtain:

$$\begin{aligned} \mathbb{E}_t[P(w^t) - D(\alpha^t)] \leq& (1-\theta)(P(w^{t-1}) - D(\alpha^{t-1})) + \frac{1}{n}\sum_{i=1}^n (\phi_i(A_i^\top w^t) - (1-\theta)\phi_i(A_i^\top w^{t-1})) \\ &- \frac{1}{n}\sum_{i=1}^n \langle A_i^\top w^t - (1-\theta)A_i^\top w^{t-1}, \nabla\phi_i(A_i^\top w^t)\rangle \\ &+ \frac{\theta}{n}\sum_{i=1}^n \left(\langle \nabla\phi_i(A_i^\top w^t), A_i^\top w^t\rangle - \phi_i(A_i^\top w^t)\right) \\ =& (1-\theta)(P(w^{t-1}) - D(\alpha^{t-1})) + \frac{1-\theta}{n}\sum_{i=1}^n (\phi_i(A_i^\top w^t) - \phi_i(A_i^\top w^{t-1})) \\ &- \frac{1-\theta}{n}\sum_{i=1}^n \langle A_i^\top w^t - A_i^\top w^{t-1}, \nabla\phi_i(A_i^\top w^t)\rangle \\ =& (1-\theta)(P(w^{t-1}) - D(\alpha^{t-1})) \\ &+ \frac{1-\theta}{n}\sum_{i=1}^n \left[\phi_i(A_i^\top w^t) - \phi_i(A_i^\top w^{t-1}) + \langle A_i^\top w^{t-1} - A_i^\top w^t, \nabla\phi_i(A_i^\top w^t)\rangle\right] \\ \leq& (1-\theta)(P(w^{t-1}) - D(\alpha^{t-1})), \end{aligned}$$

where the last inequality follows from the convexity of ϕ_i.

8 Experimental Results

In [29] and [28], the reader can find an extensive list of popular machine learning problems to which Prox-SDCA can be applied. Sharing the same primal-dual formulation, our algorithm can also be specified and applied to those applications, including Ridge regression, SVM, Lasso, logistic regression and multiclass prediction.

We focus our numerical experiments on the L2-regularized linear SVM problem with smoothed hinge loss or squared hinge loss. These problems are described in detail in Section 8.1. The three main messages that we draw from the numerical experiments are:

- Importance sampling does improve the convergence for certain datasets;
- Quartz specialized to serial samplings is comparable to Prox-SDCA in practice;
- The theoretical speedup factor is an almost exact predictor of the actual speedup (in terms of iteration complexity).

We performed the experiments on several real world large datasets, of various dimensions n, d and sparsity. The details of the dataset characteristics are provided in Table 4. In all our experiments we used Option I, which we found to be better in practice.

Dataset	# Training size n	# features d	Sparsity (# nnz/(nd))
astro-ph	29,882	99,757	0.08%
CCAT	781,265	47,236	0.16%
cov1	522,911	54	22.22%
w8a	49,749	300	3.91%
ijcnn1	49,990	22	59.09%
webspam	350,000	254	33.52%

Table 4: Datasets used in our experiments.

8.1 Applications

Smooth hinge loss with L_2 regularizer. We specify Quartz to the linear Support Vector Machine (SVM) problem with smoothed hinge loss and L_2 regularizer:

$$\min_{w\in\mathbb{R}^d} \quad P(w) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} \phi_i(y_i A_i^\top w) + \lambda g(w),$$

where

$$\phi_i(a) = \begin{cases} 0 & a \geq 1 \\ 1-a-\gamma/2 & a \leq 1-\gamma \\ \dfrac{(1-a)^2}{2\gamma} & \text{otherwise.} \end{cases}, \quad \forall a \in \mathbb{R} \tag{54}$$

and

$$g(w) = \frac{1}{2}\|w\|^2, \qquad w \in \mathbb{R}^d. \tag{55}$$

Here $y_i \in \{\pm 1\}$ is the label of the example $A_i \in \mathbb{R}^d$. One can get rid of the labels by redefining A_i to $y_i A_i$. Note that ϕ_i defined by (54) is $1/\gamma$-smooth and g defined by (55) is 1-strongly convex. In this special case, Option I in Algorithm 1 has a closed form solution:

$$\Delta\alpha_i^t = \max\left\{-\alpha_i^{t-1}, \min\left\{1-\alpha_i^{t-1}, \frac{1-y_i A_i^\top \bar{\alpha}^{t-1} - \gamma\alpha_i^{t-1}}{v_i/(\lambda n)+\gamma}\right\}\right\}.$$

See also [28, Section 5.6].

Squared hinge loss with L_2 regularizer. We now specify Quartz to the linear Support Vector Machine (SVM) problem with squared hinge loss based and L_2 regularizer:

$$\min_{w\in\mathbb{R}^d} \quad P(w) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^n \phi_i(y_i A_i^\top w) + \lambda g(w),$$

where

$$\phi_i(a) = \frac{([1-a]_+)^2}{2\gamma}, \qquad a \in \mathbb{R} \tag{56}$$

and

$$g(w) = \frac{1}{2}\|w\|^2, \qquad w \in \mathbb{R}^d. \tag{57}$$

Note that ϕ_i defined by (56) is $1/\gamma$-smooth and g defined by (57) is 1-strongly convex. In this special case, Option I in Algorithm 1 has a closed form solution:

$$\Delta\alpha_i^t = \max\left\{\frac{1-y_i A_i^\top \bar{\alpha}^{t-1} - \gamma\alpha_i^{t-1}}{\gamma + v_i/(\lambda n)}, -\alpha_i^{t-1}\right\}.$$

See also [37, Section 5.3].

8.2 Quartz and SDCA for uniform and importance sampling

In this section we compare four algorithms:

- Quartz-U: Quartz specialized to uniform serial sampling;
- Prox-SDCA [29, 28]: proximal stochastic dual coordinate ascent with uniform sampling;
- Quartz-IP: Quartz specialzed to importance sampling;
- Iprox-SDCA [37]: proximal stochastic dual coordinate ascent with importance sampling;

on three datasets: cov1, w8a and ijcnn1. We consider the $L2$-regularized linear SVM problem using squared hinge loss, as described in Section 8.1. The value of γ is set to be 1 and the value of λ varies between the datasets: 10^{-5} for w8a and ijcnn1 and 10^{-6} for cov1, whose number of training examples n is 10 times larger than the other datasets. The results are shown in Figure 3.

Utility of importance sampling. If we compare Quartz-U with Quartz-IP, it is clear that importance sampling provides better convergence rate than uniform sampling on the datasets that we tested.

Similarity between Quartz-IP and Iprox-SDCA. In all the experiments, Quartz-IP shows an almost identical convergence behaviour to that of Iprox-SDCA.

Conservative primal update in Quartz. While Quartz-IP has the same practical convergence rate as Iprox-SDCA, Quartz-U appears to be somewhat slower than Prox-SDCA in practice. One possible explanation is that the primal update in Quartz,

$$w^t = (1-\theta)w^{t-1} + \theta \nabla g^*(\bar{\alpha}^{t-1}), \tag{58}$$

is too conservative. Indeed, since the optimal solution satisfies $w^* = \nabla g^*(\bar{\alpha}^*)$, larger θ leads to faster convergence on the primal problem when the dual variable α^{t-1} is close to the optimal solution α^*. To confirm this, we tested two more aggressive primal update rules: Quartz-10θ and Quartz-100θ which change the primal update to:

$$w^t = (1-10\theta)w^{t-1} + 10\theta \nabla g^*(\bar{\alpha}^{t-1}),$$

and

$$w^t = (1-100\theta)w^{t-1} + 100\theta \nabla g^*(\bar{\alpha}^{t-1}),$$

respectively. The results are displayed in Figure 3(d), 3(e), 3(f), 3(g), 3(h) and 3(i). It is clear that with just a slightly more aggressive primal update rule than the one sanctioned by our theory, Quartz-U achieves similar practical convergence as Prox-SDCA. Recall that the primal update in Prox-SDCA is $w^t = \nabla g^*(\bar{\alpha}^{t-1})$. Notice also that the parameter θ defined by (29) is less than $1/n$, hence close to 0. Therefore, there is still a difference in the primal update rules between Quartz-100θ and Prox-SDCA.

8.3 Mini-batch experiments

In this section we demonstrate that the *theoretical speedup factor* of Quartz specialized to sampling $\hat{S}$ is a very good predictor of the *practical speedup factor*, defined as:

$$\frac{\#\text{ of iterations till } \epsilon \text{ primal dual gap is found by Quartz specialized to serial uniform sampling}}{\#\text{ of iterations till } \epsilon \text{ primal dual gap is found by Quartz specialized to } \hat{S}}.$$

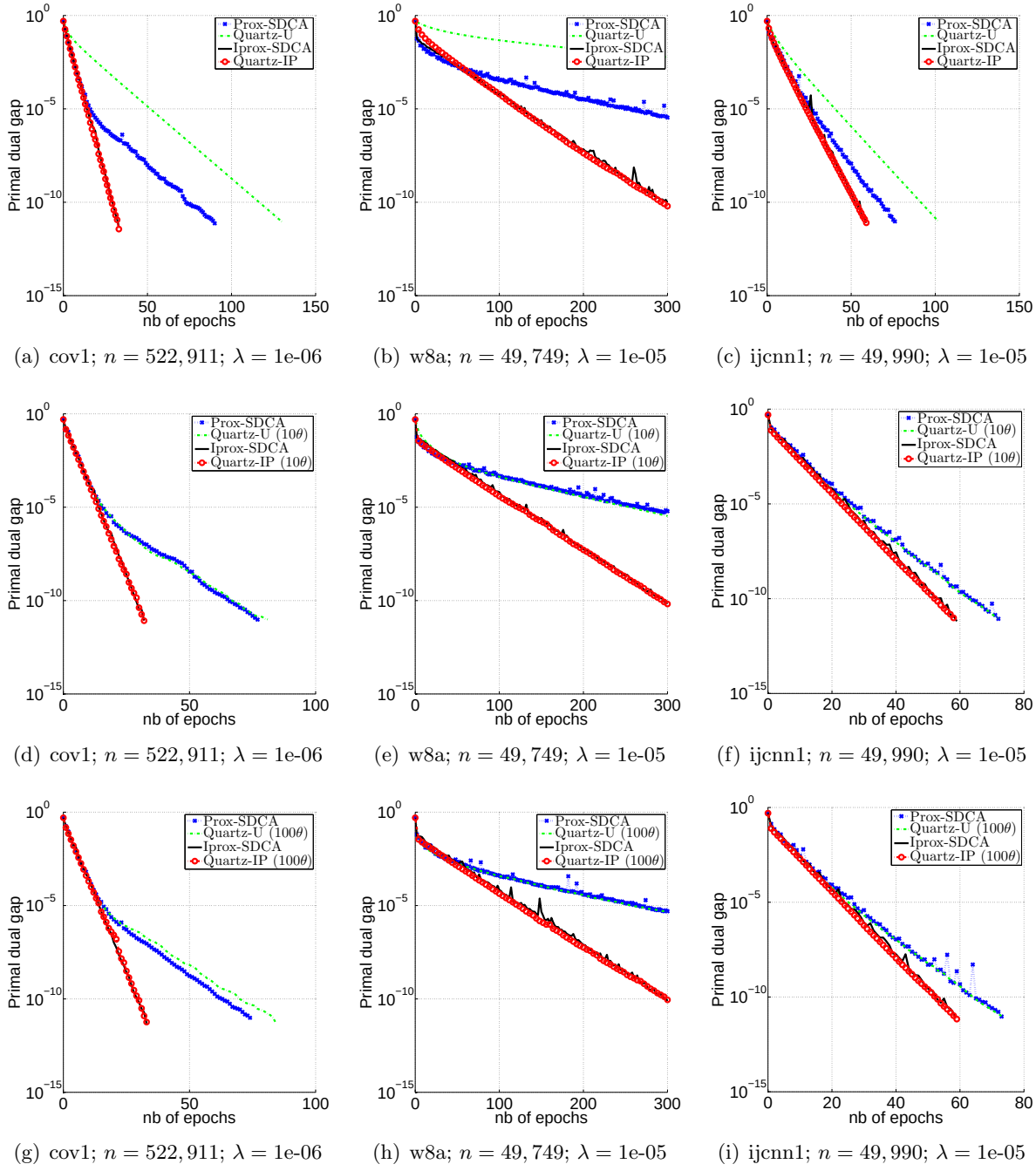
We focus on the problem of training $L2$-regularized linear SVMs with smoothed hinge loss ($\gamma = 1$), described in Section 8.1. In the experiments we chose $\epsilon = 10^{-11}$.

In Figure 4 we plot the speedup factors for Quartz specialized to the τ-nice sampling on three different datasets: astro_ph, CCAT and cov1, and for several values of λ. We observe that the practical speedup factor follows the theoretical prediction. Moreover, note that the largest λ that we choose for each dataset is to have roughly

$$\frac{\lambda\gamma n}{\max_i A_i^\top A_i} = \sqrt{n},$$

so that linear speedup is reached for all $\tau \leq \sqrt{n}$, regardless of data sparsity.

In Figure 5 we present contour lines of the theoretical and practical speedup factors, for Quartz specialized to the (c,τ)-nice sampling on the webspam dataset. We believe it is remarkable that the theoretical predictions are so accurate. Moreover, recall from the discussion in Section 6.2 that $T(c,\tau)$ is almost constant along the contour lines of $c\tau$; this is why we see nearly straight lines in the log-log plot. This feature is observed here for the real dataset also.

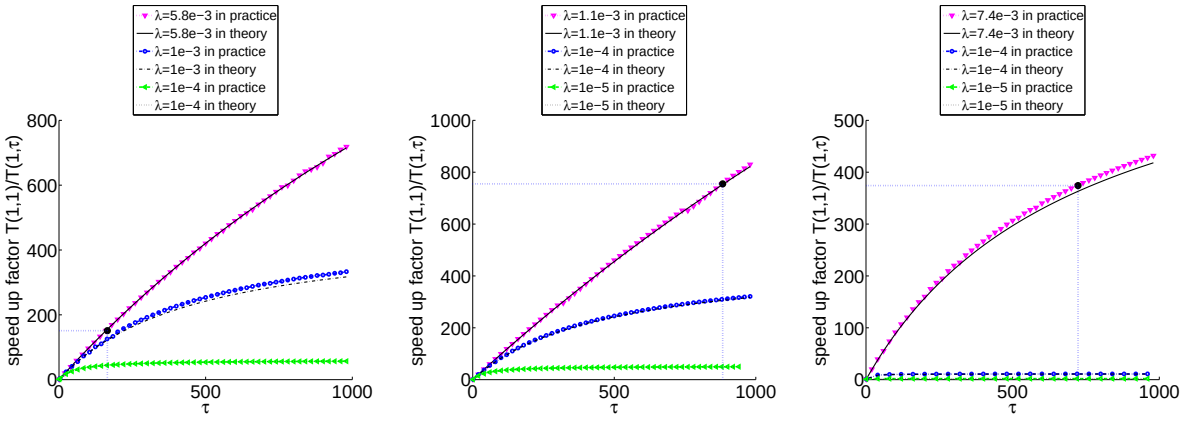


(a) cov1; $n = 522,911$; $\lambda =$ 1e-06 (b) w8a; $n = 49,749$; $\lambda =$ 1e-05 (c) ijcnn1; $n = 49,990$; $\lambda =$ 1e-05

(d) cov1; $n = 522,911$; $\lambda =$ 1e-06 (e) w8a; $n = 49,749$; $\lambda =$ 1e-05 (f) ijcnn1; $n = 49,990$; $\lambda =$ 1e-05

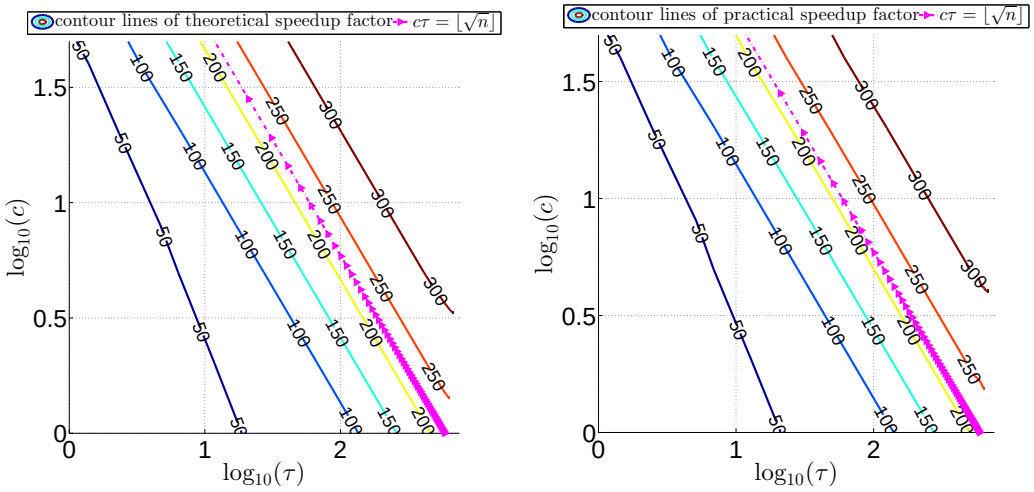
(g) cov1; $n = 522,911$; $\lambda =$ 1e-06 (h) w8a; $n = 49,749$; $\lambda =$ 1e-05 (i) ijcnn1; $n = 49,990$; $\lambda =$ 1e-05

Figure 3: Comparison of Quartz-U (uniform sampling), Quartz-IP (optimal importance sampling), Prox-SDCA (uniform sampling) and Iprox-SDCA (optimal importance sampling). In Figure 3(d), 3(e) and 3(f), we used aggressive primal update: $w^t = (1 - 10\theta)w^{t-1} + 10\theta\nabla g^*(\bar{\alpha}^{t-1})$. In Figure 3(g), 3(h) and 3(i), we used aggressive primal update: $w^t = (1-100\theta)w^{t-1}+100\theta\nabla g^*(\bar{\alpha}^{t-1})$. The loss function is the squared hinge loss. The regularizer is the L_2-regularizer.



(a) astro_ph; sparsity: 0.08%; (b) CCAT; sparsity: 0.16%; (c) cov1; sparsity: 22.22%;

Figure 4: Plots of theoretical and practical speedup factors as a function of τ, for selected values of λ. Problem: $L2$-regularized linear SVM with smoothed hinge loss and $\sigma = 1$. Datasets: astro_ph has $n = 29,882$ training samples, CCAT has $n = 781,265$ training samples and cov1 has $n = 522,911$ training samples.



(a) theoretical speedup factor. data: webspam; n=350,000; sparsity: 33.51%

(b) experimental speedup factor. data: webspam; n=350,000; sparsity: 33.51%

Figure 5: Contour plots of theoretical (Figure 5(a)) and practical (Figure 5(b)) speedup factor. The data set used is webspam. The loss function used in the smoothed hinge loss with $\sigma = 1$.

9 Conclusion

In this paper we have developed and analyzed a novel stochastic primal-dual algorithm—Quartz—for solving problems (1) and (2). This is the second stochastic method which allows an arbitrary sampling (see [22]) and the first primal-dual stochastic method with arbitrary sampling. This flexibility allows for many interesting variants of Quartz, including serial, parallel and distributed versions. The distributed variant of Quartz is the first distributed SDCA-like method with strong theoretical convergence bounds.

In Table 5 we highlight selected characteristics of existing primal-dual stochastic methods.

Algorithm	Serial uniform sampling	Serial optimal (importance) sampling	τ-nice sampling	Arbitrary sampling	Additional data-dependent speedup	Direct primal-dual analysis	Acceleration
SDCA [29]	✓	✗	✗	✗	✗	✗	✗
ASDCA [28]	✓	✗	✓	✗	✗	✗	✓
AccProx-SDCA [28]	✓	✗	✗	✗	✗	✗	✓
DisDCA [34]	✓	✗	✓	✗	✗	✗	✗
Iprox-SDCA [37]	✓	✓	✗	✗	✗	✗	✗
APCG [15]	✓	✗	✗	✗	✗	✗	✓
SPDC [36]	✓	✓	✓	✗	✗	✓	✓
Quartz	✓	✓	✓	✓	✓	✓	✗

Table 5: Summary of selected characteristics of stochastic primal-dual algorithms.

Unlike some of the existing methods, our method is not accelerated. We leave the development of an accelerated Quartz method for future research.

References

[1] Alekh Agarwal and Leon Bottou. A lower bound for the optimization of finite sums. *arXiv:1410.0723*, 2014.

[2] Joseph K. Bradley, Aapo Kyrola, Danny Bickson, and Carlos Guestrin. Parallel coordinate descent for L1-regularized loss minimization. In *28th Int. Conf. on Machine Learning*, 2011.

[3] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Math. Imaging and Vision*, pages 120–145, 2011.

[4] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *arXiv:1407.0202*, 2014.

[5] Olivier Fercoq, Zheng Qu, Peter Richtárik, and Martin Takáč. Fast distributed coordinate descent for minimizing non-strongly convex losses. *IEEE International Workshop on Machine Learning for Signal Processing*, 2014.

[6] Olivier Fercoq and Peter Richtárik. Accelerated, parallel and proximal coordinate descent. *arXiv:1312.5799*, 2013.

[7] Olivier Fercoq and Peter Richtárik. Smooth minimization of nonsmooth functions by parallel coordinate descent. *arXiv:1309.5885*, 2013.

[8] Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear svm. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 408–415, New York, NY, USA, 2008. ACM.

[9] Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S Sathiya Keerthi, and S Sundararajan. A dual coordinate descent method for large-scale linear svm. In *In ICML 2008*, pages 408–415, 2008.

[10] Martin Jaggi, Virginia Smith, Martin Takáč, Jonathan Terhorst, Sanjay Krishnan, Thomas Hofmann, and Michael I. Jordan. Communication-efficient distributed dual coordinate ascent. In *Advances in Neural Information Processing Systems 27*. 2014.

[11] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems 26*, pages 315–323. 2013.

[12] Jakub Konečný, Jie Lu, Peter Richtárik, and Martin Takáč. mS2GD: Mini-batch semi-stochastic gradient descent in the proximal setting. *arXiv:1410.4744*, 2014.

[13] Jakub Konečný, Zheng Qu, and Peter Richtárik. S2CD: Semi-stochastic coordinate descent. Technical report, University of Edinburgh, 2014.

[14] Jakub Konečný and Peter Richtárik. S2GD: Semi-stochastic gradient descent methods. *arXiv:1312.1666*, 2014.

[15] Qihang Lin, Zhaosong Lu, and Lin Xiao. An accelerated proximal coordinate gradient method and its application to regularized empirical risk minimization. Technical Report MSR-TR-2014-94, July 2014.

[16] Julien Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. Technical report, 10/2014.

[17] Indraneel Mukherjee, Kevin Canini, Rafael Frongillo, and Yoram Singer. Parallel boosting with momentum. Technical report, 2013.

[18] Ion Necoara and Andrei Patrascu. A random coordinate descent algorithm for optimization problems with composite objective function and linear coupled constraints. *Computational Optimization and Applications*, 57:307–337, 2014.

[19] Yurii Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

[20] Yurii Nesterov. Gradient methods for minimizing composite objective function. *Math. Programming, Ser B*, 140:125–161, 2013.

[21] Peter Richtárik and Martin Takáč. Distributed coordinate descent method for learning with big data. *arXiv:1310.2059*, 2013.

[22] Peter Richtárik and Martin Takáč. On optimal probabilities in stochastic coordinate descent methods. *arXiv:1310.3438*, 2013.

[23] Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization problems. *arXiv:1212.0873*, 2012.

[24] Peter Richtárik and Martin Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Math. Programming*, 144:1–38, 2014.

[25] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *arXiv:1309.2388*, 2013.

[26] Shai Shalev-Shwartz and Ambuj Tewari. Stochastic methods for ℓ_1-regularized loss minimization. *Journal of Machine Learning Research*, 12:1865–1892, 2011.

[27] Shai Shalev-Shwartz and Tong Zhang. Proximal stochastic dual coordinate ascent. Technical report, 2012.

[28] Shai Shalev-Shwartz and Tong Zhang. Accelerated mini-batch stochastic dual coordinate ascent. In *Advances in Neural Information Processing Systems 26*, pages 378–385. 2013.

[29] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss. *Journal of Machine Learning Research*, 14(1):567–599, February 2013.

[30] Martin Takáč, Avleen Bijral, Peter Richtárik, and Nathan Srebro. Mini-batch primal and dual methods for SVMs. In *30th International Conference on Machine Learning*, 2013.

[31] Rachael Tappenden, Peter Richtárik, and Burak Büke. Separable approximations and decomposition methods for the augmented lagrangian. *Optimization Methods and Software*, 2014.

[32] Rachael Tappenden, Peter Richtárik, and Jacek Gondzio. Inexact block coordinate descent method: complexity and preconditioning. *arXiv:1304.5530*, 2013.

[33] Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *arXiv:1403.4699*, 2014.

[34] Tianbao Yang. Trading computation for communication: Distributed stochastic dual coordinate ascent. In *Advances in Neural Information Processing Systems 26*, pages 629–637. Curran Associates, Inc., 2013.

[35] Tianbao Yang, Shenghuo Zhu, and Yuanqing Lin. Analysis of distributed stochastic dual coordinate ascent. *arXiv:1312.1031*, 2013.

[36] Yuchen Zhang and Lin Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. Technical Report MSR-TR-2014-123, September 2014.

[37] Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling. *arXiv:1401.2753*, 2014.